Multi-Criteria Decision Making in Software Development: A Systematic Literature Review

University of Oulu
Faculty of Information Technology and Electrical Engineering
Department of Information Processing
Master's Thesis
Yong Zhou
Date 22/06/15

Abstract

Multiple Criteria Decision Making is a formal approach to assist decision makers to select the best solutions among multiple alternatives by assessing criteria which are relatively precise but generally conflicting. The utilization of MCDM are quite popular and common in software development process. In this study, a systematic literature review which includes creating review protocol, selecting primary study, making classification schema, extracting data and other relevant steps was conducted. The objective of this study are making a summary about the state-of-the-art of MCDM in software development process and identifying the MCDM methods and MCDM problems in software development by systematically structuring and analysing the literature on those issues.

A total of 56 primary studies were identified after the review, and 33 types of MCDM methods were extracted from those primary studies. Among them, AHP was defined as the most frequent used MCDM methods in software development process by ranking the number of primary studies which applied it in their studies, and Pareto optimization was ranked in the second place. Meanwhile, 33 types of software development problems were identified. Components selection, design concepts selection and performance evaluation became the three most frequent occurred problems which need to be resolved by MCDM methods. Most of those MCDM problems were found in software design phase. There were many limitations to affect the quality of this study; however, the strictly-followed procedures of SLR and mass data from thousands of literature can still ensure the validity of this study, and this study is also able to provide the references when decision makers want to select the appropriate technique to cope with the MCDM problems.

Keywords
Multiple Criteria Decision Making (MCDM), Software Development Process, Systematic Literature Review (SLR), Analytic Hierarchy Process (AHP), Software design phase, MCDM problems

Foreword

This master thesis is the biggest English scientific work I have ever finished, the whole process of writing is full of hardships which almost lasts 10 months since last June till this April. As a consequence, amounts of experiences and knowledge were acquired during this journey, but most importantly, it enforced my endurances of loneliness and tediousness and it also taught me how to be patient, because the results of this work were generated by three members, so I had to wait the outcomes from other members very often.

During this long journey, tons of helps were received from my supervisors, my friends, my families. Many thanks to my two supervisors, Jouni Marrkula and Sandun Dasanayake, who cooperated with me to generate the results and gave me numbers of advices to write and optimize the thesis, especially when they were so busy, they still patiently guide me to move on. Special thanks to Lucy who stays with me in the same office, thank her for the help of Nvivo. In the end, I would like to thank my parents, thank them for the encouragements, especially the warm words in last harsh winter, they are my only motivation to move forward. Thank all the people accompanied with me during this journey!

Contents

1. Introduction

Software development is a series of activities which are full of uncertainties and vagueness. Through the whole process of software development, a number of decisions need to be made, for example selecting alternative solutions, evaluating candidate components, and selecting design concepts. Therefore making decision is an inevitable work for engineers or developers in software development. A reliable and rigorous decision making process can help software engineers to have better management of software development activities, such as mitigate risks and maximize profits (Falessi, Cantone, Kazman, & Kruchten, 2011). However, in many circumstances, most of the decision making problems encountered in software development are relevant to multiple criteria which need to be simultaneously considered, so attentions of multiple criteria decision making (MCDM) is gradually increased in software development in last decades. As a result, a growing number of relevant techniques used to address the MCDM problems which are occurred in software development were adopted, such as analytic hierarchy process (AHP) (Saaty, 1980), technique for order preference by similarity to ideal solution (TOPSIS) (Yoon, 1987), weight sum model (WSM) (Fishburn, 1967), and Pareto optimization (Miettinen, 1998; Hwang & Masud, 1979).

Different MCDM techniques suit different decision situations (Eldrandaly, Ahmed, & AbdelAziz, 2009). For example, AHP is suitable for those problems which are hard to quantify decision makers' preference for various criteria and alternatives (Saaty, 1980), WSM is recommended for those decision making contexts that their criteria and alternatives are in the same unit (Fishburn, 1967), and Pareto optimization is applied to handle multi-objectives decision problems (Miettinen, 1998). It stands to reason that those MCDM techniques are convenient tools for software engineers to resolve those decision making problems. However new concerns from many decision makers occurred that they have difficulties in deciding which MCDM methods should be used in software development and how those MCDM methods can be relevant to software development problems. To mitigate those concerns, an overview of MCDM in software development seems to be important for all the software engineers who are responsible for making decisions in their daily work.

For investigating the current situation of MCDM and obtaining the overview of MCDM in software development, a Systematic Literature Review (SLR) which is a research technique used to analyze the state-of-the-art in a particular field of knowledge (Kitchenham, 2004) was conducted. Consequently, the state-of-the-art of MCDM in software engineering will be summarized and analyzed. More specifically, the investigations about the state-of-the-art of MCDM in the software development process phases were conducted. Software development process are a coherent set of activities used by systems engineers and systems developers to plan for, design, build, test, and deliver information systems (Sommerville, 2004). It likes an assembly line at factories which can divide the whole software production process into several phases. For the purposes of achieving the objectives of this study, three aspects of MCDM will be considered to describe the current situation of MCDM in software development during the process of SLR, they are:

- MCDM problems which are identified during the software development process;
- Distributions of MCDM problems in different software development process phases;

- MCDM techniques used to address corresponding MCDM problems in software development process phases.

Relevant data will be extracted based on these three aspects, and through the analysis of these data, different kinds of MCDM problems occurred in software development will be summarized and classified, and their distributions in each phase of software development process will be explored. Most importantly, the MCDM techniques used to address those problems will be also identified. Considering all those results, an overview of MCDM in software development would be obtained in the end.

The rest of structures of this thesis are organized as follows. Chapter 2 will describe the background of multiple criteria decision making and a basic recognition of it can be obtained. Chapter 3 will illustrate the whole procedure of systematic literature review and discuss the considerations which should be concerned during that procedure. Chapter 4 will present the results extracted based on research questions and classify them according to their common features. Chapter 5 will make detailed analysis and discussion on the results, some interesting findings will be explained in this chapter. Chapter 6 will make a conclusion on all those research questions, discuss the limitations of this study, and propose a research direction of future work.

2. Multi-Criteria Decision Making

Decision making is a process which is quite intuitive when considers the single criterion problem in order to select the most preferable alternative (Gwo-Hshiung & Jih-Jeng, 2011). However, when decision makers need to cope with multiple criteria, especially when those criteria are conflicting with each other, the process of decision making would be much more complicated and needs more sophisticated methods in order to address the trade-offs among criteria (Gwo-Hshiung & Jih-Jeng, 2011). The concept of multi-criteria decision making was firstly introduced since the 1700s by Benjamin Franklin, who applied this concept in a simple paper system for deciding important issues (Labaree & Bell, 1956). With the development of MCDM over centuries, the understandings and definitions of it becomes more and more specific and clear.

Numbers of researchers have discussed the definitions of MCDM. For example, Belton and Stewart (2002) described that MCDM methods are tools which can help decision makers in handling the difficulties of reaching compromise or consensus between conflicting objectives and criteria. Stewart (1992) defined MCDM as a formal approach which can solve types of problem by attempting to represent a number of individual relatively precise but generally conflicting criteria. Mateo (2012) identified MCDM methods as a branch of a general class of operations research models which are applied to address complex problems which are full of high uncertainty, conflicting goals, different forms of data and information, multiple interests and criteria. Based on all those discussions of MCDM definitions, it can be defined as a formal approach to assist decision makers to select the optimal solution among multiple candidates by assessing relatively precise but generally conflicting criteria.

2.1 Categorizations of MCDM

With the advancement of MCDM, it was further divided into two main categories according to their purposes and data types, which are multiple objectives decision making (MODM) and multiple attributes decision making (MADM) (Mateo, 2012; Gwo-Hshiung & Jih-Jeng, 2011; Hwang & Yoon, 1981). Among them, MODM problems refer to problems that have infinite numbers of feasible alternatives, where the decision variables functionally relate to the objectives and constraints (Eldrandaly, Ahmed, & AbdelAziz, 2009); therefore, MODM methods designed to address those problems aim to achieve the desired goal by considering infinite numbers of continuous feasible alternatives within the given constrains on a vector of decision variables (Mateo, 2012; Gwo-Hshiung & Jih-Jeng, 2011).While MADM problems are problems that have a relatively small number of alternatives, where the alternatives are represented in terms of attributes which are explicit and are regarded as both decision variables and decision criteria (Eldrandaly, Ahmed, & AbdelAziz, 2009). In order to address MADM problems, MADM methods are designed to be applied in the evaluation facet, and be used to select the predetermined and discrete alternatives (Mateo, 2012; Gwo-Hshiung & Jih-Jeng, 2011).

MODM and MADM can be further sub-divided into two categories according to the goal preference structure of the decision makers, which refers to the decision makers' preference of evaluation criteria and/or alternatives (Malczewski, 2006). If there is a single goal-preference structure, which means there is only one goal, it can then be referred to individual decision making, regardless of how many stakeholders involved. On the other hand, if there are multiple goal preference structures with multiple stakeholders, so it is group decision making (Malczewski, 2006). Consequently,

individual multi-criteria decision making and group multi-criteria decision making can be regarded as the sub-categories of MCDM. Furthermore, decision making can be broadly classified into decisions under certainty and decisions under uncertainty. Decisions under certainty refer to decisions that are made when all relevant information about decision situation is acquired, in other words, decision makers have sufficient knowledge of the decision environment (Eldrandaly, Ahmed, & AbdelAziz, 2009). While decisions under uncertainty mean decisions are made under an unpredictable environment, two basic types of uncertainty were derived from this unpredictable decision situation:

- Uncertainty with limited information about the decision situation,
- Uncertainty with fuzziness concerning the description of the semantic meaning of the events, phenomena or statements.

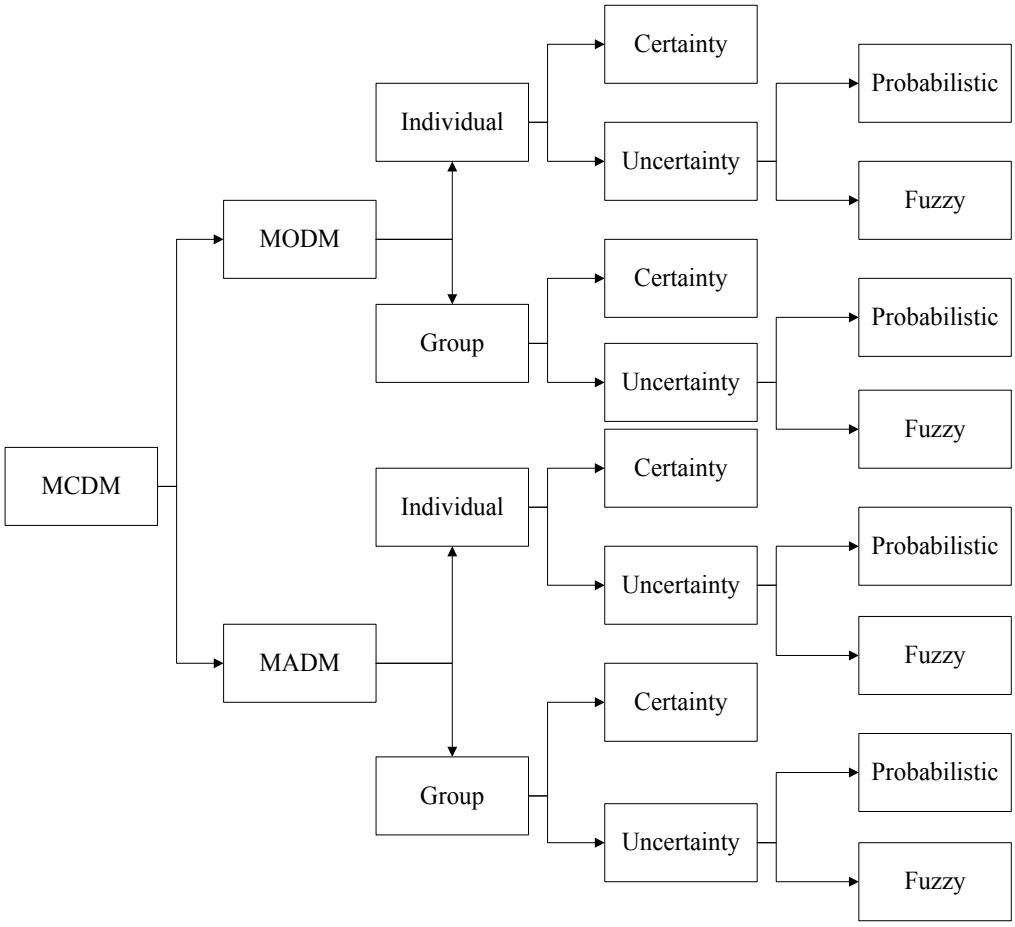

Figure 1. Categorizations of multi-criteria decision making (Malczewski, 2006)

Consequently, decision making under uncertainty can be sub-divided into probabilistic and fuzzy decision making based on its type of uncertainty (Malczewski, 2006). An increasing number of MCDM methods used in current research can be classified based on the categorizations of MCDM. For example, AHP is one of the subordinates of MADM, since it is utilized to make decisions under certainties through considering explicit and small numbers of alternatives, and all the relevant information on decision situation should be acquired by decision makers; therefore AHP is a multiple attribute decision making technique under certainty. Through the analysis of each MCDM method by following the categorizations of MCDM (Figure 1), every MCDM method can be easily understood like the previous example.

2.2 Characteristics of MCDM problems

Besides the MCDM techniques, another important element of MCDM is the MCDM problems, which are addressed by MCDM techniques. MCDM problems frequently occurred in the phases of software development process. Numbers of instances can be used to explain MCDM problem. For example, if one type of programming language is needed for the programmers to implement the software product, many evaluation criteria, such as clearness of definition, expressivity, input-output, reliability, support of modularity, and several language candidates, like Java, C, C#, C++, would come into decision makers' considerations. In the meantime, the appropriate MCDM technique is applied to concurrently manage and consider all those alternatives and the relevant criteria. In the end, the best decision can be made to select the most appropriate programming language. This example is a typical MCDM problem which would occurred in the early phase of software development process. However, the characteristics of MCDM problem are more than that.

The MCDM problems always relate to multiple objectives and attributes, and numbers of different stakeholders are always the sources of those different objectives and attributes, so multiple stakeholders become one of the MCDM problem characteristics (Belton & Stewart, Multiple Criteria Decision Analysis: an integrated approach, 2002). During the process of MCDM, the final decision maker needs to consider all the opinions from the stakeholders, and tries to reach a consensus or compromise especially when those opinions are mutually conflicting. Therefore, the analysis of problem may need to be conducted within groups which involving representatives of all stakeholders, or they can do it separately within sub-groups, but all the considerations need to be gathered to the final decision maker at the end. Except that, another characteristic can be defined as that the alternatives or considerations of MCDM problem may be very large or infinite. It is impossible to satisfy all of them, so one or some more detailed phases or parts of the problem can be focused, a short list of alternatives can be created for the decision makers to select (Belton & Stewart, Multiple Criteria Decision Analysis: an integrated approach, 2002).

2.3 Detailed Steps of MCDM process

After introducing MCDM techniques and MCDM problems, the connections between them need to be described based on the detailed process of MCDM (Figure 2). Generally, the process of resolving MCDM problems encompasses eight steps (Dodgson, Spackman, Pearman, & Phillips, 2009). First, the decision context needs to be established, which targets to establish the goals of MCDM and identify decision makers and other stakeholders. Besides that, it also needs to consider the context of the appraisal. Second, the alternatives which are the candidates needed to be evaluated in the process of MCDM should be identified. Third, the criteria for assessing the consequences of each alternative need to be identified, then those criteria need to be organized by classifying them under higher-level and lower-level objectives in a hierarchy. For example, if the goal of MCDM is building a house, under the objective of cost, the corresponding criteria can be cost of implementation, cost of preparation, cost of maintenance and other relevant costs. Whether the objective is higher-level or lower-level, it depends on your assessments on the trade-offs among different objectives. Fourth, the expected performance of each alternative against the criteria needs to be assessed, then the value associated with the consequences of each alternative for each criterion also needs to be assessed.

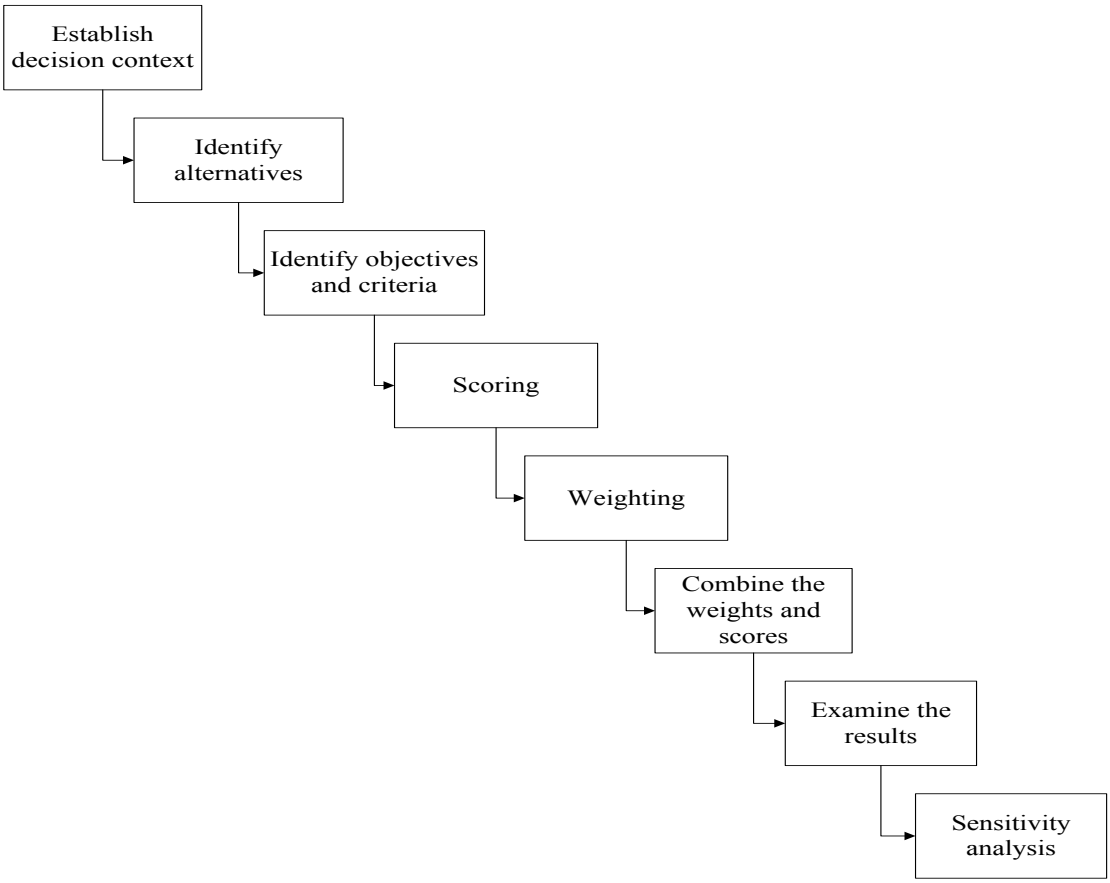

Figure 2. Detailed steps of MCDM process (Dodgson, Spackman, Pearman, & Phillips, 2009)

Fifth, weights for each of the criteria to reflect its relative importance to the decision are assigned. The relative importance between criteria means the range of difference of the alternatives, and how much that difference matters (Dodgson, Spackman, Pearman, & Phillips, 2009). Sixth, the weights and scores for each alternative are combined to calculate an overall value. Different MCDM techniques have different algorithms to derive the overall value, for example, in AHP, the overall preference score on each alternative is simply the weighted average of its scores on all the criteria (Saaty, 1980). After the calculation of the overall value on each alternative, a ranking list of alternatives will be generated. Seventh, the obtained results need to be examined, if they are unexpected results, the decision makers need to determine whether they agree the way forward or make other recommendations. Last, sensitivity analysis is conducted for examining the influence caused by the vagueness about the inputs or disagreements between people to the final overall results (Dodgson, Spackman, Pearman, & Phillips, 2009). The first three steps are basically the same among different MCDM techniques, what different are the steps of scoring, weighting and combine the weights and scores, which are the most important steps in the MCDM process.

3. Research Method

The research method used in this research is systematic literature review (SLR) which is a means of identifying, evaluating and interpreting all available research literature relevant to a particular research question, or topic area, or phenomenon of interest (Kitchenham, 2004). Seven steps are contained during the SLR process, first, review protocol should be developed which encompasses rationale for the review, research questions, inclusion and exclusion criteria, keywords used for searching for literature, quality assessment procedures and other relevant elements, thus the area of relevant literature can be targeted (Kitchenham, 2004); after the scope of research is outlined, the retrieval of literature based on the review protocol can be started; when the initial retrieval is completed, a huge amount of relevant papers will be obtained.

Next step is the process of primary studies selection which is the most time and efforts consuming, it can be conducted by excluding and including papers according to the predefined protocol. Once the original primary studies are gathered, the assessment of study quality needs to be done, as a result, the final primary studies can be obtained; as soon as the final primary studies are ready, data extraction can be immediately carried on by following the predefined research questions; after data extraction, the original data without deep analysis can be acquired; and in order to generate the results those original data will be synthesised; at the end of SLR, the results acquired from data synthesis will be demonstrated in the form of report (Figure 2). In this thesis, the process of SLR is utilized to summarize MCDM techniques and address some issues related to MCDM in software development process phases.

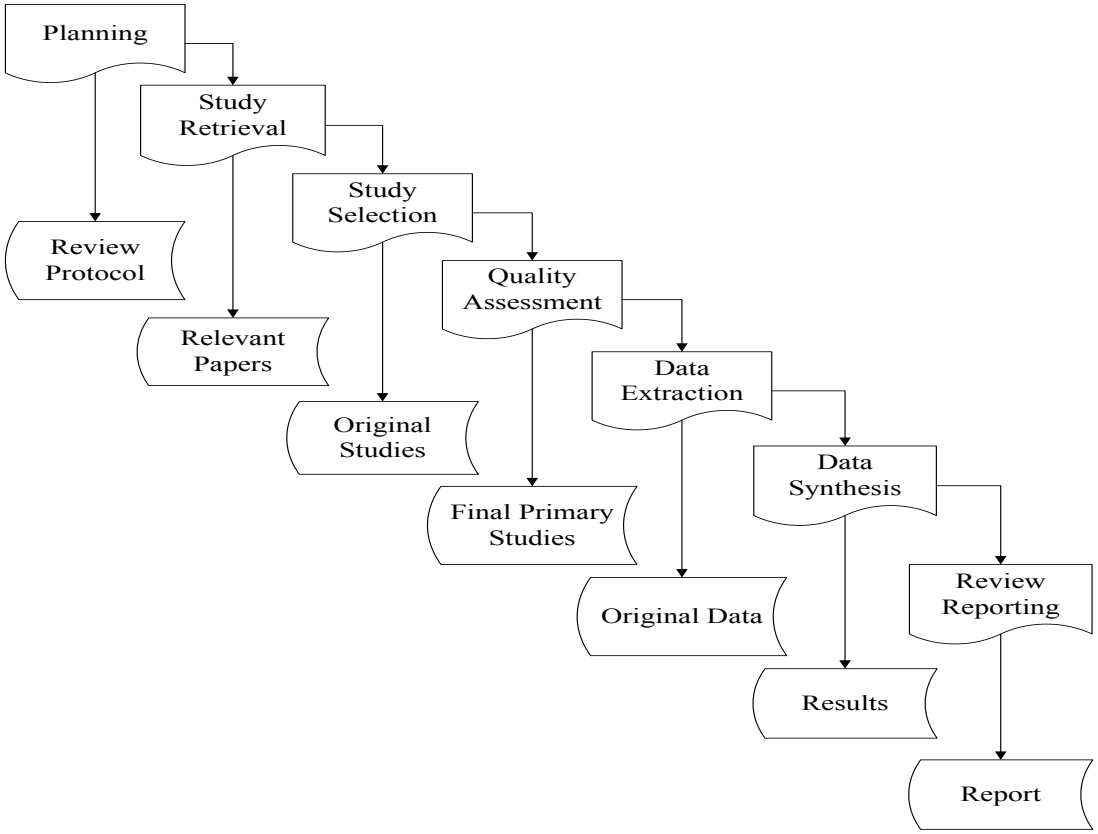

Figure 3. Systematic Literature Review process (Kitchenham, 2004).

3.1 Review Protocol

The methods used to conduct a specific systematic review are specified by a review protocol which is necessary to decrease the possibility researcher bias (Kitchenham, 2004). A review protocol contains research questions, search strategy which includes search terms and data sources, inclusion and exclusion criteria, quality assessment procedures, data extraction strategy and data synthesis strategy. All the elements of review protocol will be presented.

3.1.1 Research Questions

The purpose of this study is to make a summary of the state-of-the-art on multiple criteria decision making in software development process phases, so three research questions were defined based on discussion among members and current understandings of this area. Most importantly, they were constructed based on the objectives of this thesis. The research questions are as follows:

- RQ1: What are the MCDM problems in software development?
- RQ2: How are the MCDM problems distributed during the software development process?
- RQ3: What are the MCDM techniques utilized in software development?

With regard to RQ1, the MCDM problems occurred in software development will be identified, and they will be classified into different categories based on their common features, and the most occurred problems would be identified. For RQ2, the distributions of all those MCDM problems in different software development process phases will be presented, and the phases have the most MCDM problems would be identified. In regard to RQ3, a summary about the MCDM techniques used to address those problems which are summarized from the first research question will be obtained, and the most frequented used techniques in software development will be identified from the summary. After the research questions were defined, the scope of retrieval of studies can be basically targeted.

3.1.2 Search Strategy

According to the research questions, two general phrases which are decision making and software engineering were identified. After that, online dictionary was used to find synonyms of those terms and different phrases with the same meaning. Table 1 shows the final used keywords for retrieval.

Table 1. Search Terms.

Category	**Keywords**
Decision making	decision-making
	making decision
	decision support
	decision analysis
Software engineering	software development
	software design
	software engineering
	system development

	system design
	system engineering

For this systematic literature review, six desired databases were selected to be the sources of literature. In order to determine the databases, an analysis of the digital databases' utilization on 88 example papers related to SLR was conducted, the relevant statistics then suggested us to select the top six databases. Those digital databases are Scopus, IEEE Xplore, ACM Digital Library, Science Direct, Web of Science and ProQuest. The search strings constructed for literature retrieval are different based on the different retrieval rules of each database, they can be checked in the appendix A.

3.1.3 Inclusion and Exclusion Criteria

One of the most important elements of review protocol is the inclusion and exclusion criteria, which defines the instructions of how to select papers. The whole selection of papers should strictly follow the criteria, otherwise there will be huge differences on the selected papers between you and your partners which simultaneously do the selection, and thus the selected papers make no sense and it would also waste your time to eliminate the differences.

There were three rounds for us to select the desired papers, each round had its inclusion and exclusion criteria, and the later round was conducted on the foundation of the former round (Figure 3). The title, keywords and metadata were reviewed in the first round. The second round was excluding papers by reading the abstracts. Introduction and conclusion parts were reviewed to identify the final relevant papers in the third round. The selection process which are about choosing desired papers followed those inclusion criteria and exclusion criteria in Table 2.

Table 2. Inclusion & Exclusion Criteria.

<table>
<tr><th colspan="3">Selection Process</th><th>Inclusion Criteria</th><th>Exclusion Criteria</th></tr>
<tr><td rowspan="3">1st Round (Title)</td><td rowspan="2">2nd Round (Abstract)</td><td rowspan="3">3rd Round (Full-Text)</td><td>• It is about decision-making in software or system engineering domain,
• It is related to at least one of the software or system development life cycle phases,
• It is peer-reviewed journals,
• It is conference articles,
• It is book chapters,
• It is doctoral dissertations.</td><td>• It is not written in English,
• It is an opinion paper,
• It is a presentation,
• It is an interview,
• It is a summary / extended abstract,
• It is a technical report,
• It is an introductory chapters for conference proceedings,
• It is an editorial.</td></tr>
<tr><td>• It describes decision making technique or process or method or procedure or protocol in the development of software or system.</td><td>• Only mentioned decision making techniques, but not related to software or system development.</td></tr>
<tr><td></td><td>• It describes multi-criteria or multi-objectives or multi-attributes decision analysis or decision making or decision support,
• Do not mention the multi-criteria decision making, but it used the MCDM techniques directly, like AHP, TOPSIS.</td><td>• Only mentioned decision making on software or system development, but it didn't actually address or deal with the problem with decision making techniques.</td></tr>
</table>

3.2 Study Selection Process

Five rounds of paper selection were illustrated in Figure 4 which includes initial search, round 0 to 3. After initial search, a set of initial papers were obtained, it is inevitable that there would be duplicates existed among databases, so removing all those duplicates before papers selection was instantly carried on. First, the literature from initial search was imported to Refworks which is a web-based bibliography management software package, duplicates among different databases were then detected and they were finally deleted within Refworks.

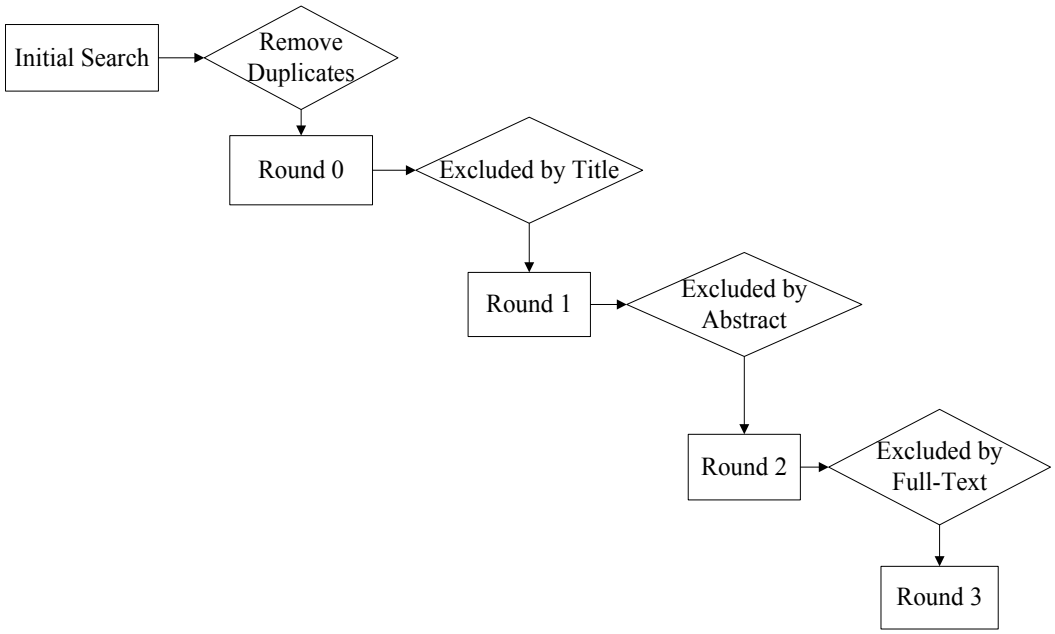

Figure 4. The Paper Selection Process.

After removing the duplicates, three rounds of selections were conducted. Firstly, the references without duplicates were exported from Refworks to Excel spreadsheets, and each piece of reference contains detailed information which includes titles, authors, publishers, abstracts, publication years and links. When the selection of papers had begun, there were three options for SLR researchers to choose and make a mark to determine the paper was included or not. The three options are 0, 1 and 3. 0 means rejected, 1 means accepted, while 3 means not sure. Nevertheless when two researchers combined their results, there were numbers of combinations corresponded to different meanings (Table 3). At the beginning of the selection, two researchers did the selection separately, after the selection, the Excel spreadsheet with conflicting papers was delivered to the third researcher, and the third researcher then made the final decision in order to determine whether the conflicting one should be included or not, a final spreadsheet with relevant information on selected studies was given in the end (Figure 5).

Table 3. Using Number to Select Papers.

Number	Scenarios	Results
0	0+0	Rejected
1	0+1, 1+0	Not Sure
2	1+1	Accepted

3	0+3, 3+0	Rejected
4	1+3, 3+1	Accepted
6	3+3	Accepted

There was a problem occurred during the process of papers selection, which was the misunderstanding on the same inclusion and exclusion criteria, in other words, how to have the same comprehensions of inclusion and exclusion criteria with your partners. This type of problem occurred at the beginning of our selection, the selection criteria were accepted by SLR researchers in different meanings. As a result, amounts of time was wasted to discuss the conflicting papers, what was worse, the selection of desired studies from ten thousand papers needed to be redone. Different understandings of the inclusion and exclusion criteria indicate that the review protocol is not accurate and good, because it causes ambiguity. In order to avoid this problem, a method was applied with the ongoing of selection process used to measure the reliability of selection, the name of this method is Cohen's kappa which is a statistic measure of reliability for qualitative items (Carletta, 1996). The agreement between two raters who each classify *N* items into *C* mutually exclusive categories (Cohen, 1960) was measured by Cohen's kappa.

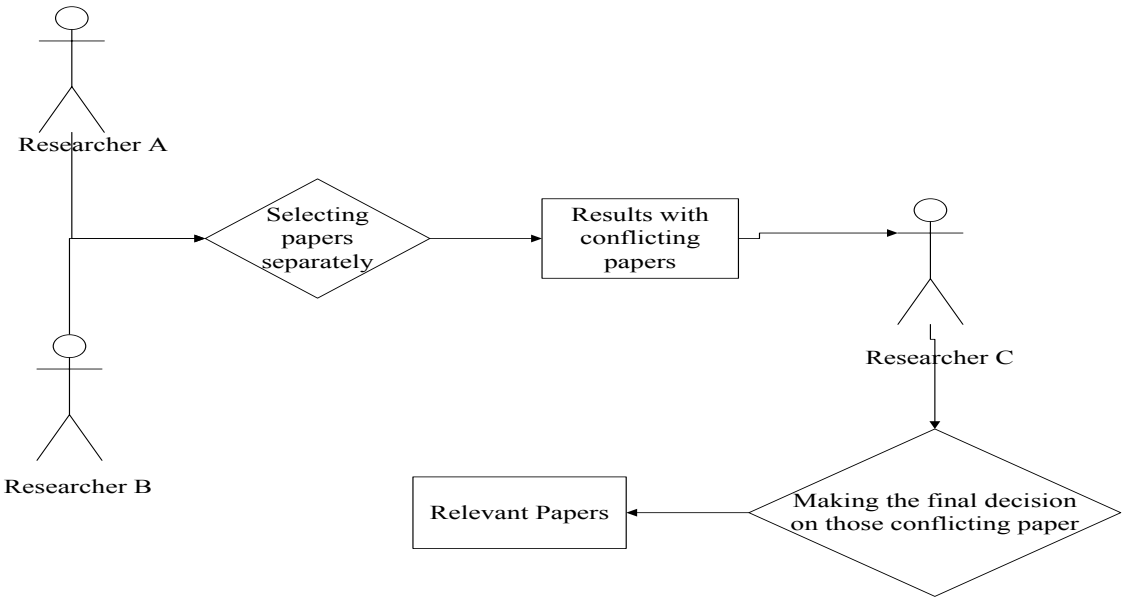

Figure 5. The Selection of Papers among Multiple Researchers.

3.3 Quality Assessment

After the selection of literature, the original primary studies complied with inclusion and exclusion criteria were identified, however, a deep-level evaluation was conducted to select the final primary studies. In order to assess the quality of studies, this section aims to identify the potential risks which would influence the quality and validity of primary studies. A set of quality criteria which are the detailed version of inclusion and exclusion criteria were created to evaluate each primary study. The quality criteria can be the means to weight the importance of individual studies when results are being synthesized (Kitchenham, 2004).

Quality criteria were used to deeply evaluate the original primary studies, which are different with inclusion and exclusion criteria. The quality criteria used in this study are

composed of three questions, the quality standards were derived from those questions, and each standard has its corresponding score. The evaluation was separately conducted between two SLR researchers, the scores obtained from two researchers were then combined, and in the end, the average score was calculated to decide if this primary study is included or not. Here are those three questions:

- Does the study have clear MCDM problems?
- Does the MCDM problem occur during the software development process?
- Does the study contain the steps of decision making process?

Four standards derived from the questions are then listed as follows:

- Study has clear MCDM problems and clear steps of decision making (score=3, excellent quality);
- Study has unclear MCDM problems and clear steps of decision making (score=2, good quality);
- Study has clear MCDM problems and unclear steps of decision making (score=2, good quality);
- Study has unclear MCDM problems and unclear steps of decision making (score=1, fair quality);
- Study doesn't have MCDM problems and steps of decision making (score=0, bad quality).

The mathematical algorithm of average score is:

$$\text{avg.} = \sum_{i=1}^{n} s\,/n \tag{1}$$

In this algorithm, avg. is the average score of each study after the quality assessment of SLR researchers, n represents the number of SLR researchers, and s represents the score of each study given by each researcher. As a result, the average score was obtained by adding all the scores, and the sum of scores was then divided by the number of researchers, the final result was the average score of this study. If the average score of one primary study were less than 2, it would be excluded, otherwise, it is the final primary study.

3.4 Classification Schema

The objective of creating a classification scheme is sorting the primary studies and classifying them with different keywords. In our study, before Nvivo was used to help researchers to review all the primary studies, different nodes were created in Nvivo based on research questions, the abstracts of primary studies were then reviewed again to verify if the classification covered all the primary studies. There were four major categories, and numbers of subcategories below those major categories.

The domain is the first category created in Nvivo, it stands for the domain of the primary studies, for example, the paper talked about the embedded system, so the domain of it is embedded system. The MCDM method is the second category, each MCDM method was extracted from its corresponding primary study, and it was then placed below the category of MCDM method, the name of its node is the name of the MCDM method, primary studies with the same method were grouped into one same node. Each primary study had one or multiple MCDM methods, and one method was also applied in multiple primary

studies, so one primary study could be under different nodes, and one node could also contain different primary studies. The third category is the MCDM problem, the problems were extracted from primary studies first, and they were initially all under the node of MCDM problem, based on their common attributes, they were then grouped into six subcategories. The last category is software development process phases, all the MCDM problems were classified into different software development process phases depending on their behaviours.

3.5 Data Extraction

Data extraction is a process of reviewing of those selected primary studies. In this process, Nvivo was utilized to assist researchers to do the review of primary studies. Here are five steps to describe the process of data extraction with Nvivo:

- Step 1: Importing the selected primary studies into Nvivo, and renaming them for ease of management;
- Step 2: Creating the nodes based on the predefined classification schema;
- Step 3: Starting to read the primary studies with the created nodes;
- Step 4: Dragging the relevant information to the corresponding nodes;
- Step 5: After reviewing all the primary studies, categorizing all the nodes based on their same attributes.

If from your point of view that just one round of data extraction were not enough to have a good understanding of the primary studies, memos in Nvivo could be created to help you summarize each primary study, and the name of memo should be as same as the primary studies.

3.6 Synthesis of findings

Data synthesis involves collating and summarizing the results of the included primary studies (Kitchenham, 2004). The descriptive synthesis was applied in our study, that is to say, tables and texts were used to describe the results extracted from primary studies. In this study, three classifications were identified based on the predefined research questions, which were MCDM techniques, software development problems, software development process phases.

The first part named MCDM techniques referred to the MCDM techniques applied during software development process phases. Their abbreviations, full names and references which means the corresponding primary study applied this technique in its study were summarized. After that, the top 7 of MCDM techniques were selected to be analyzed, and its corresponding MCDM problems and references were grouped into one table. The second part called software development problems meant the MCDM problems which were addressed by MCDM techniques, those problems occurred every phase of software development process. The works done by us were classifying them into different types, for example, the type of software component, all the MCDM problems which are related to software components were then gathered into this type. There was also a table which contains the name of the problem and corresponding references.

The third part was called software development process phases which represented the distribution of MCDM problems occurred in each software development process phase. MCDM problems were summarized and categorized into different phase based on its characteristics, the comparison between them and the features of software development process phases. Tables were also created to show the results which contained MCDM

problems and their corresponding references. The last part called domains referred to the domains of the primary studies. Because the summarized results didn't make any sense, so this part was removed without further illustration and discussion.

3.7 Validity Assessment

During the process of SLR, there would be lots of bias which were caused by human subjective behaviors (Kitchenham, 2004), their identifications were needed for the validity assessment of this study.

Selection bias

When the selection of papers was carried on, some papers may be excluded just because of the lack of relevant knowledge. The way to mitigate this threat is leaving the unclear paper to the third researcher, because the third researcher has much more knowledge and experiences than the other two researchers. Another threat would occur when the predominance of the opinion of one researcher over the other (Nicolò, Carmine, Michael, & Tony, 2014), if two researchers do the selection together. This threat was taken into our considerations at the beginning of the selection, the selection was separately conducted, when one period of works was done, the results from different researchers would be combined and those conflicting selection would be delivered to the third researcher (Figure 4). The threat can also come from researcher's personal subjective judgment, the method applied to eliminate the gaps is Cohen's Kappa which can ensure the reliability of the selections.

Search term bias

When the keywords were identified, but the synonymous of the keywords were found in online dictionary are not enough to cover all the relevant papers, in this condition, some biases would be caused. The way to decrease this bias should be consulting others who have much more knowledge about the keywords than you, and they know other phrases which have the same meaning, or belong to the same category. For example, "decision making" is the key phrase used in this study, if relevant information were not obtained, the similar phrases like "decision analysis" or "decision support" might be overlooked. Hence, it is very important to have relevant knowledge on these search terms.

Measurement bias

When the titles, abstracts or even the introductions of the paper are reviewed, there are still risks to miss the potential information. On the contrary, even though all the key phrases are mentioned in the abstracts of some papers, they still might not provide the desired information. When the primary studies were measured, not only the key phrases were needed to be retrieved, but whether the information was required or not was also needed to be identified. For example, the title of one of the original papers is "An Intelligent Knowledge-based Multiple Criteria Decision Making Advisor for Systems Design", it contained all the key phrases, and when its abstract was reviewed, it also related to the software development, but it was still excluded, because it described a tool which helped the designer to select MCDM methods, not about the problems in software development.

4. Results

This chapter will present the results after the whole process of SLR which involves selection results and review results. As introduced in Chapter 3, the selection process of primary studies consists of six rounds, and each round removed irrelevant studies according to the inclusion and exclusion criteria. The first round was initial search with search strings in titles, abstracts or keywords, 18039 papers were retrieved. After that, about 30% studies were decreased after removing duplicates, the total number of papers became 12698. The biggest drop of papers amount occurred after round 1, almost ten thousand papers were excluded, and thus the number of papers is 2521. The following two rounds which select studies based on abstracts and full-texts made the amount of papers become 118 which is 0.65% of the literature from initial search. In round 4, the quality assessment based on quality criteria was conducted, as a result, the number of studies became 56, which were the final primary studies (Table 4).

Table 4. Number of literature each round.

Database Name	Initial	Round 0	Round 1	Round 2	Round 3	Round 4
Scopus	10848	8267	1198	184	54	18
IEEE Xplore	951	853	312	117	17	13
ACM	220	215	97	45	7	5
Science Direct	482	474	143	49	11	9
Web of Science	2227	979	297	76	11	4
ProQuest	3311	1910	474	121	18	7
Total	18039	12698	2521	592	118	56
Percentage	100%	70.36%	13.98%	3.28%	0.65%	0.31%

Most of the primary studies were published during the last ten years (Figure 6), it indicates that MCDM in software development is a popular topic for the software scientific research; therefore, a systematic summary of those literature is important to gain the clarity of the decision making in software engineering. The distribution of selected paper types among the primary studies can also indicate that this study is of high quality, because 32 literature are conference papers and 24 literature are journal articles. The results summarized from them can mostly represent the state-of-the-art of MCDM in software development.

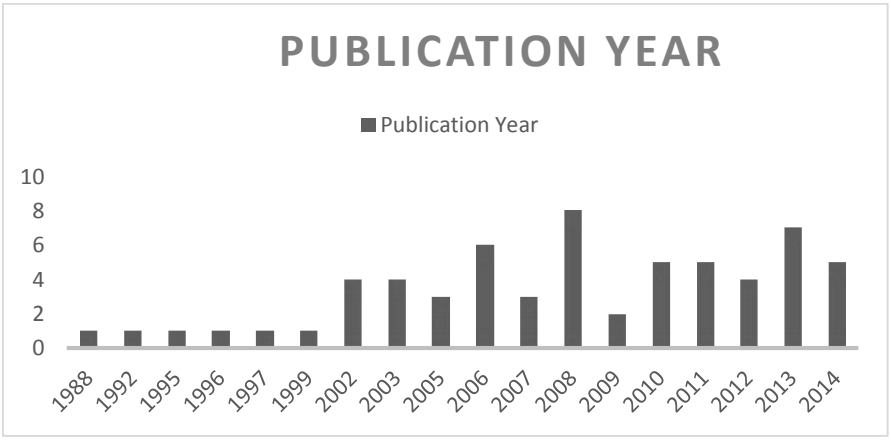

Figure 6. The distributions of Publication Year.

After the selection of primary studies, the review works about data extraction were immediately carried on. Relevant data was extracted based on the predefined research questions, so there were three sections about the review results. The first section was about the results of the first research question, all the software development problems identified from primary studies were summarized and classified into different categories based on their common features, and every category will be then stated in details along with a table which contained the members of software development problems which belonged to this category. The second section aimed to group those software development problems into different software development process phases, each phase and its corresponding problems will be briefly introduced. The third section was about the results of the third research question which was about the utilization of MCDM techniques, it will present the overall results in tables, and then introduce the top 7 MCDM techniques in details which included the corresponding software development problems addressed by them.

4.1 Software Development Problems (RQ1)

The structure of each primary study is that one or several problems occurred during the development of software need to be addressed by one or several MCDM methods, so each paper has its own software development problems. After the data extraction, 33 types of problems occurred in software development were obtained and they were categorized into six classifications (Figure 7). 44.7% (17.9%+26.8%) of software development problems are the problems of design concepts which are about the selection and analysis of the best design alternatives and the problem of software methods which are about the selections and analysis of tools and techniques used in software development. The problems of software quality attributes evaluation also have a high percentage among software development problems which is about 17.9%. The discussion about software requirements is a little bit less than software quality and it is about 16.1%. The problems related to software components are about 14.3%, and the least percentage of problems are those problems related to project management which is about 8.9%.

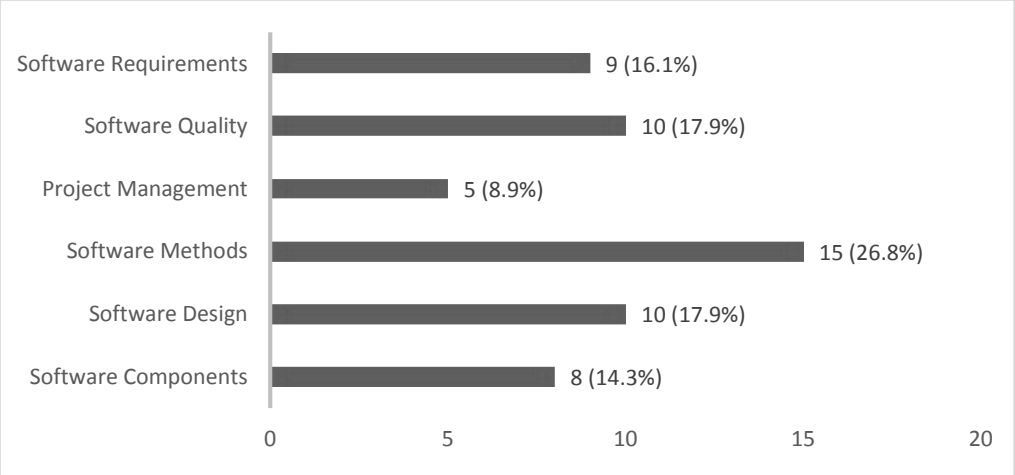

Figure 7. Problems in software development.

Software Components

An individual software component is a software package, a web service, a web resource or a module which encapsulates a set of functions or data, and it can communicate with other components via interfaces (Niekamp, 2005). In the early phase of software development, the selection, evaluation and measurement of components for system configuration based on multiple design attributes are crucial and necessary steps for software designers. There were 8 primary studies which used MCDM to conduct the selection, evaluation and measurement of all kinds of software components (Table 5), and four types of MCDM problems of software components were categorized, they were component selection (P15, P29, P39, P58, and P74), component evaluation (P44), release planning (P53) and component partitioning (P21).

Components selection is the most frequent occurred MCDM problem in this category, this problem was solved by concurrently considering multiple design attributes (P15, P29, P39, P58, and P74). Besides the selection of components, the evaluation of components was also solved by simultaneously considering multiple attributes. For release planning, the reasons of why this problem belonged to this category might confuse readers, but the essence of this problem was defining weighting factors for component modifiability to design release plans based on thresholds for the relative extent of modifiability acceptable for a release, so it was classified to software components (Omolade & Guenther, 2005) (P53). The last problem is the components partitioning, which moved the focus of partitioning problems into a multi-criteria perspective (Gaetana, Toberiu, & Ivica, 2013)(P21).

Table 5. Problems related to software component.

No.	MCDM Problems	References
1	Components selection	[P15][P29][P39] [P58][P74]
2	Components evaluation	[P44]
3	Release Planning	[P53]
4	Components partitioning	[P21]

Software Design

Software design is the process which aims to create a specification of a software artifact by an agent, intended to accomplish goals, using a set of primitive components and subject to constraints (Paul & Yair, 2009). There were many aspects needed to be considered in the design of software; therefore many issues were related to the decision making. Furthermore, because of the multiple design considerations, like compatibility, extensibility, reliability, reusability, the multi-criteria decision making problems frequently occurred. From table 6, it shows decision making problems in software design were mostly related to the selection of system design concepts summarized from our study (P2, P7, P54, P65, and P67), besides it, the decision making problems were the selection of candidate layouts and operational design (P20), design space exploration in the design of embedded system (P2, P14), design optimization (P62, P69), and simulation of system design (P69).

System design concepts selection is the most frequent occurred problem in this category. Design concepts were generated and evaluated, and they were then selected by MCDM methods in the earliest design stages (P2, P7, P54, P65, and P67). For the candidate

layouts and operational designs selection, it provided an approved ability to perform a multi-objective comparison of several candidate layouts and operational designs, and then selected the most appropriate layouts and designs (Fatah, Kash, & Andrea, 2012) (P20). Another problem in this category was the design optimization which was resolved by evaluating the impact of alternative system designs on high level goals and finding optimal design options among the alternatives (P62, P69). The MCDM problem could also happen in the simulation of system design which was achieved by estimating the levels of goal satisfaction contributed by alternative system design (William & Emmanuel, 2011) (P69). The last MCDM problem of software design was design space exploration which was domain-specific exploration problems in embedded systems design (P2, P14).

Table 6. Problems related to software design.

No.	MCDM Problems	References
1	Selection of candidate layouts and operational designs	[P20]
2	Design Optimization	[P62][P69]
3	Design space exploration	[P2][P14]
4	Selection of system design concepts	[P2][P7][P54][P65][P67]
5	Simulation of system design	[P69]

Software Methods

The software development problems of software methods which were about the selections of technologies, methods and models were summarized in this category. All of them were MCDM problems in which different goals, objectives, or attributes, criteria should be considered. All the 11 primary studies were related to selection (Table 7). For example, in primary study 45, it was about the selection of software architecture styles, different candidate architecture styles, and different objectives were then simultaneously considered, at last a ranking list was generated for the developers to make a choice. Eight MCDM problems were summarized in this part from our systematic literature review. They were customization of software engineering technologies (P31), model evaluation, comparison and selection (P56), selection of configuration items (P6, P35, and P46), selection of data mining algorithms (P73), selection of software architecture styles (P45), selection of software development life cycle model (P37, P48), selection of software development tools (P51, P41), and selection of system alternatives (P24, P61, P70, and P71).

With regard to the customization of software engineering technologies, its MCDM problem actually was about the selection of the attribute weighting heuristics (Jingzhou & Guenther, 2008) (P31). For model evaluation, comparison and selection, its aim was achieved by comparing five aerodynamics tools across fifteen key criteria in a typical context (Rajabally & Holywell, 2006)(P56). In terms of the problem of configuration items selection (P6, P35, and P46), because many items on configuration management can be defined as configuration items, such as hardware, software, documentation, so the contexts of this problem were different, as a result, the methods used to cope with them were also different. In our study there were three different MCDM methods used to cope with them, which are SAF (P46), DBD (P6), and OWA (P35). For the problem of data algorithms selection (P73), it was solved by experimentally comparing the performance of several popular data mining algorithms through using different performance metrics over public domain software defect datasets from the repository (Yi, Gang, Guoxun, Wenshuai, & Honggang, 2010).

For software architecture styles selection, it was a MCDM problem in which different goals and objectives should be considered. While software development life cycle models selection is similar with the software architecture styles selection which is considered as evaluating the specific needs and challenges of a project and then choosing the most appropriate model (P45, P37, and P48). For the problem of tools selection in software development, it utilized the relevant factors based on the most common features as the selection criteria in ranking the software tools (P51, P41). The last problem was system alternatives selection which was also based on multi-criteria decision analysis to compare alternative systems (P24, P61, P70, and P71).

Table 7. Problems related to software method.

No.	MCDM Problems	References
1	Customization of software engineering technologies	[P31]
2	Model evaluation, comparison and selection	[P56]
3	Selection of configuration items	[P6][P35][P46]
4	Selection of data mining algorithms	[P73]
5	Selection of software architecture styles	[P45]
6	Selection of software development life cycle model	[P37][P48]
7	Selection of tools	[P51][P41]
8	Selection of system alternatives	[P24][P61][P70][P71]

Project Management

Software project management is the art and science of planning and leading software projects which include the management of software development process and project planning, monitoring and control (Andrew & Jennifer, 2005). Issues of multi-criteria decision making in project management were various, about four types of MCDM problems were identified in our study (Table 8). In primary study 4 and 8, the MCDM problems were the allocations of software development works which aimed to give decision supports for distributed task allocations for global software companies that regarded to multiple criteria.

The MCDM problem in primary 43 was about the resource allocation by assigning weights to software development activities, then identifying the most important activity with those weights, at last allocating the resources based on the ranking of activities (Mindy, Hoang, & Xuemei, 1999). In primary study 68, the MCDM problem was optimizing the schedule of a software project by considering multiple parameters, such as the project duration, the work discontinuities of development teams in successive iterations and the release (delivery) time of software deliverables with a multi-objectives linear programming technique (Vassilis & Pandelis, 2007). The last MCDM problem obtained from our study was the selection of control laws among a number of alternative control law structure in primary study 18 which aimed to minimize the cost of implementation (Doug & Marc, 1988).

Table 8. Problems related to project management.

No.	MCDM Problems	References
1	Allocation of software development works	[P4][P8]
2	Resource Allocation	[P43]

3	Optimization of schedule	[P68]
4	Selection of control laws	[P18]

Software Quality

Two related but different notions were used to define software quality, one was software functional quality which reflects how well it complies with or conforms to a given design, based on functional requirements or specifications, the other one was software structural quality which refers to how it meets non-functional requirements that support the delivery of functional requirements, such as performance, robustness, reliability, usability, uncertainty (Pressman, 2005). There were 10 primary studies which applied MCDM techniques to cope with its MCDM problems, 6 out of 10 primary studies were about the evaluation of software quality, and all of them were related to evaluation and measurement of software structural quality (Table 9).

If the MCDM problems were software quality evaluation, they were basically about a decision maker chooses the best alternative that satisfies the evaluation criteria among a set of candidate solutions (P1, P3, P5, P25, P47, P55, and P72). The most occurred problem in the category of software quality was performance evaluation which aimed to measure system performance which involved the tradeoffs between multiple, potential conflicting criteria (P5, P25, P47, P55, and P72). In primary study 32, the problem was optimal scenarios selection which takes customer demand uncertainty as a noise factor to identify an optimal scenario from alternative designs (Jiunn-Chenn, Taho, & Cheng-Yi, 2010), because it involved demand uncertainty which belonged to the scope of software quality, so it was classified into this category.

Primary study 36 was similar with primary study 32, which was about usability patterns selection, while its selection was based on the identification of the order of attractiveness of a list usability patterns, then allowing the selection of the most appropriate pattern in new communication resource (Kenia, Hildeberto, & Elizabeth, 2006). Costs of software development also belongs to the scope of software quality, so primary study 16 was included in this category, which was about cost estimation in software development. Primary study 3 was about the evaluation of quality trends, which was different with the evaluation of quality, the overall trend of quality was evaluated during the evolution of software systems by considering the releases of a project as units to be ranked (Alexander & Emmanouil, 2013).

Table 9. Problems related to software quality.

No.	MCDM Problems	References
1	Cost Estimation	[P16]
2	Evaluation of software quality	[P1]
3	Evaluation of quality trend	[P3]
4	Performance Evaluation	[P5][P25][P47][P55][P72]
5	Selection of optimal scenarios	[P32]
6	Selection of usability patterns	[P36]

Software Requirements

Software requirement is a field within software engineering which copes with fulfilling the demands of users or stakeholders which are to be resolved by software (Radatz,

Geraci, & Katki, 1990). The activities in software development process which were related to software requirement can be broadly divided into four phases, elicitation, analysis, specification and management (Bourque, Dupuis, Abran, Moore, & Tripp, 1999). Most of the decision making problems which were related to software requirement occurred frequently during the four phases (Table 10).

In the phase of requirement elicitation, the MCDM problem could be the evaluation of early requirements which were about the functionalities the software should provide, the quality requirements it should satisfy (P19, P60). It could also simply be the process of requirement elicitation which the MCDM techniques are used to calculate the weights versus each criterion, then aggregate the performance of each software component (Chi-Yo, Hsiang-Chun, Gwo-Hshiung, & Hong-Yuh, 2010) (P12). The selection of requirement engineering (RE) techniques was a MCDM problem in the period of requirement elicitation, MCDM method provided supports for RE to tailor or define RE process models and selected the most appropriate RE techniques for a specific project (Li & Armin, 2003)(P40). In the phase of requirement analysis, the MCDM problems were requirements tradeoffs analysis which aimed to analyze tradeoffs among requirements when multiple alternative design solutions satisfy different requirements to some extent (P22, P60, and P66). It could also be the risk assessment of software requirement which assessed the absolute and relative importance rates and then determined the priorities of these risk factors (P26, P63). The last MCDM problem summarized in this part was requirement negotiation (P28), requirement negotiation doesn't belong to a specific phase in a project, but should be used from the start to the end of the whole software development life cycle (Boehm, et al., 1998).

Table 10. Problems related to software requirement.

No.	MCDM Problems	References
1	Evaluation of early requirements	[P19][P60]
2	Requirements elicitation	[P12]
3	Requirements negotiation	[P28]
4	Requirements tradeoff analysis	[P22][P60][P66]
5	Selection of requirement engineering (RE) techniques	[P40]
6	Risk assessment	[P26][P63]

4.2 Software Development Process Phases (RQ2)

After obtained the results of software development problems, the distributions of them need to be done based on their behaviours. As mentioned in Chapter 1, software development process is a coherent set of activities that leads to the production of a software product (Sommerville, 2004). Although there are many different software processes, some fundamental activities are common to all software processes (Sommerville, 2004):

- Software specification, which defines the functionalities of the software and constraints on its operations,
- Software design and implementation, which aims to realize the predefined software specification,
- Software validation, which confirms the validity of that the software meets the customer requirements,

- Software evolution, which defines how to maintain the software to meet the changing customer needs.

Two of the four fundamental activities were divided into several phases. Among them, software specification was divided into feasibility study and requirement definition, software design and implementation encompassed two phases which are software design and software implementation. Each phase has different numbers of software development problems, the details of it were all presented in Figure 8. It could be clearly recognized that software development problems occurred most frequent in the phase of software design. The distribution of software development problems in the phases of feasibility study and software evolution were basically the same. While in the phases of system implementation and software validation, none of the software development problem was addressed. Exceptionally, two problems belonged to all phases which means it can occur in every phase. Since one primary study can have multiple MCDM problems, so the number of MCDM problems is more than the number of primary studies.

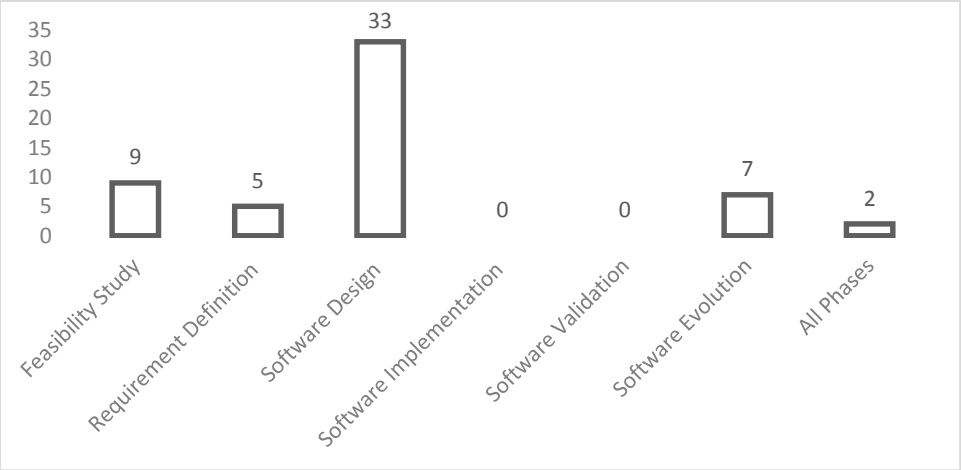

Figure 8. Distribution of problems in software development process phases.

Feasibility Study

A feasibility study is conducted at the beginning of the software development process, carrying out a feasibility study includes information assessment, information collection and report writing (Sommerville, 2004). It always aims to answer three questions:

- Does the system contribute to the overall objectives of the organization?
- Can the system be developed using current technologies and within existing budgetary constraints?
- Can the system be integrated with other systems which are already in place?

From our study, 9 primary studies used MCDM methods to resolve corresponding problems in this phase (Table 11). Cost estimation was conducted when the developers started to analyse the cost and benefit (P16) which provided a summary analysis of the benefits of the project compared with the cost. Evaluation of quality trend and risk assessment belonged to the task of risk analysis (P3, P26, and P63). Optimization of schedule (P68) was done when software engineers conducted the task of high level work breakdown structure and schedule. The allocation of software development works and resource allocation were also part of the project definition phase (P4, P8, and P43).

Table 11. Problems distributed in feasibility study phase.

No.	MCDM Problems	References
1	Cost estimation	[P16]
2	Evaluation of quality trend	[P3]
3	Allocation of software development works	[P4][P8]
4	Resource Allocation	[P43]
5	Selection of software architecture styles	[P45]
6	Risk assessment	[P26][P63]
7	Optimization of schedule	[P68]

Requirement Definition

Requirement definition phase can be divided into three sub-phases which are requirements elicitation and analysis, requirements specification, requirements validation (Sommerville, 2004). Requirement elicitation and analysis is the process of gathering the requirements of system through knowledge of existing systems, consulting with potential users and procurers, task analysis and other relevant channels. Requirement specification is the activity of translating information into a document which defines a set of requirements. Requirements validation is the process to validate those requirements for realism, consistency and completeness (Sommerville, 2004). The MCDM problems in this phase were mostly related to requirement (Table 12). For example, the evaluation of early requirements (P19, P60) which was about the evaluation of the functionalities the software should provide, the quality requirements it should satisfy. Requirement elicitation (P12) which the MCDM techniques were used to calculate the weights on each criterion, then aggregate the performance of each software component. Requirement trade-offs analysis (P22, P60, and P66) which aimed to analyze tradeoffs among requirements when multiple alternative design solutions satisfied different requirements to some extents. Optimization of schedule (P68) which was optimizing the schedule of a software project by considering multiple parameters with a multi-objectives linear programming technique (Vassilis & Pandelis, 2007).

Table 12. Problems distributed in requirement definition phase.

No.	MCDM Problems	References
1	Evaluation of early requirements	[P19][P60]
2	Requirements elicitation	[P12]
3	Requirements tradeoff analysis	[P22][P60][P66]

Software Design

A software design is a description of the software structure, the data which is part of the system, the interfaces between system components and the algorithm used (Sommerville, 2004). Therefore, the design process activities will involve architectural design, abstract specification, interface design, component design, data structure design and algorithm design. The essence of software design is actually making decisions about the logical organization of the software, so there are amounts of multi-criteria decision making problems, plenty of solution alternatives, design concepts, candidate components need to be selected in this phase.

18 different kinds of MCDM problems from 33 primary studies were summarized which are related to analysis and design after our systematic literature review (Table 13). 12 MCDM problems were about selecting the candidate designs or tools, like selection of candidate layouts (P20), system design concepts ([P2], [P7], [P54], [P65], and [P67]), system alternatives ([P24], [P61], and [P70]), configuration items ([P6], [P35], and [P46]), components ([P15], [P29], [P39], [P58], [P74], and [P21]), optimal scenarios (P32), control laws (P18), software development life cycle models (P37, P48), requirement engineering techniques (P40), selection of usability patterns (P36), and model evaluation, comparison and selection (P56). Besides those problems relate to selection, there were 6 kinds of problems relate to analysis and evaluation, such as design optimization (P62, P69), design space exploration (P2, P14), simulation of system design (P69), customization of software engineering technologies (P31), evaluation and analysis alternative system (P71), and components evaluation (P44).

Table 13. Problems distributed in software design phase.

No.	MCDM Problems	References
1	Selection of candidate layouts and operational designs	[P20]
2	Selection of system design concepts	[P2][P7][P54][P65][P67]
3	Selection of components	[P15][P29][P39][P58][P74]
4	Selection of optimal scenarios	[P32]
5	Selection of configuration items	[P6][P35][P46]
6	Selection of system alternatives	[P24] [P61][P70]
7	Selection of control laws	[P18]
8	Selection of software development life cycle model	[P37][P48]
9	Selection of requirement engineering (RE) techniques	[P40]
10	Selection of tools	[P41][P51]
11	Selection of usability patterns	[P36]
12	Model evaluation, comparison and selection	[P56]
13	Design Optimization	[P62][P69]
14	Design space exploration	[P2][P14]
15	Simulation of system design	[P69]
16	Customization of software engineering technologies	[P31]
17	Evaluation and analysis alternative system	[P71]
18	Components evaluation	[P44]

Software Implementation and Software Validation

Software implementation is naturally carried on behind the system design processes. The main works of software implementation are basically programming and debugging, there is no problem related to decision making, so software development problem needed to be solved by decision making was not detected in this phase. Software validation involves testing processes to inspect and review whether the system complies with its specification and meets users' expectations (Sommerville, 2004). Three types of testing process are included in the software validation which is component testing, system testing and acceptance testing. All the activities related to testing are not relevant to decision making, so none of the software development problems detected in this phase need to be addressed by decision making methods.

Software Evolution

Software evolution, which is also called software maintenance, is a process to ensure the flexibility of software system, and it is much cheaper when changes are made to software compared with changes to system hardware, so software evolution is a phase to maintain software to adapt to the changes of users' needs (Sommerville, 2004). Three types of MCDM problems were identified in this phase through the SLR (Table 14). They are evaluation of software quality (P1) which was conducted in the sustainment phase to examine the quality of software (Ahmad, Mohsen, & Farshad, 2010), performance evaluation (P5, P25, P47, P55, and P72) which aimed to simultaneously consider all the conflicting performance criteria and it needed great prudent efforts, and selection of data mining algorithms (P73) which aimed to evaluate software reliability by applying data mining techniques in software engineering data to identify software defects or defaults (Yi, Gang, Guoxun, Wenshuai, & Honggang, 2010).

Table 14. Problems distributed in software evolution phase.

No.	MCDM Problems	References
1	Evaluation of software quality	[P1]
2	Performance Evaluation	[P5][P25][P47][P55][P72]
3	Selection of data mining algorithms	[P73]

All Phases

There are two exceptions of the MCDM problems which doesn't belong to any specific phase in a project, but should be used from the start to the end of the whole software development life cycle. One of the MCDM problems was requirement negotiation (P28) which was not a one-time episode in a project, and it was used early on and repeated in later stage, new stakeholders and new objectives had to be considered often leads to negotiation in each cycle (Boehm, et al., 1998; Grünbacher & Seyff, 2005). As a consequence, requirement negotiation belonged to all phases. The other problem is release planning (P53) which addressed decisions related to selecting and assigning features to create a sequence of consecutive product releases that satisfies important technical, resources, budgets, and risk constraints (Ruhe & Saliu, 2005). In agile development, release planning focused on planning for the next iteration, products was produced each phase in agile development, so the release planning was conducted in all the software development process phases.

Table 15. Problems distributed in all phases.

No.	MCDM Problems	References
1	Requirement Negotiation	[P28]
2	Release planning	[P53]

4.3 MCDM techniques (RQ3)

The third research question is about the utilization of MCDM techniques used to address the software development problems, which were summarized based on the first research question. After data extraction, 33 kinds of MCDM techniques were summarized from 56 primary studies. From the observations of Table 16, some phenomena were identified

that one technique can be applied by multiple studies, and one study can also utilize multiple techniques which combined as hybrid methodologies.

Table 16. MCDM techniques in software development.

No.	Acronym	Full Name	References
1	AHP	Analytic Hierarchy Process	[P3][P15][P24][P37] [P39][P41][P43][P48] [P51][P53][P60][P62] [P70][P71][P74]
2	Pareto	Pareto Optimization	[P2][P7][P14][P19] [P20][P31][P69] [P72][P18]
3	TOPSIS	Technique for Order Preference by Similarity to Ideal Solution	[P32][P48][P67][P72] [P73]
4	WSM	Weighted Sum Model	[P18][P21][P66]
5	OWA	Ordered Weighted Averaging	[P26][P35][P56]
6	QFD	Quality Function Deployment	[P16][P47][P65]
7	Outranking	PROMETHEE and ELECTRE	[P24][P73][P31]
8	MAUT	Multi-Attribute Utility Theory	[P41][P54]
9	SAW	Simple Additive Weight	[P12][P63]
10	SAF	Sensitivity Analysis Framework for Optimum Expansion Planning	[P46]
11	ANP	Analytic Network Process	[P71]
12	Bokhari	Bokhari Algorithm	[P8]
13	CBR	Case-Based Reasoning	[P40]
14	CFPR	Consistent Fuzzy Preference Relations	[P63]
15	DBD	Decision-Based Design framework	[P6]
16	DEMATEL	Decision Making Trial and Evaluation Laboratory	[P44]
17	ESTEVAL	ESTimation and EVAluation	[P29]
18	ESM	Even Swap Method	[P22]
19	FCM	Fuzzy Clustering Means	[P2]
20	FMCDM	Fuzzy multiple criteria decision making	[P1]
21	FCI	Fuzzy Choquet Integral	[P45]
22	LRPs	Multi-objective linear programming scheduling mode	[P68]
23	MACBETH	Measuring Attractiveness through a Category Based Evaluation TecHniques	[P36]
24	MCDA	Multi-criteria decision analysis model	[P4]
25	MDQ	Mechatronic Design Quotient	[P58]
26	MPARN	Multi-Criteria Preference Analysis Requirements Negotiation	[P28]
27	OBCL	Outcome Based Control Limits	[P55]
28	OTSO	off-the-shelf option	[P39]
29	PVA	Performance Value Analysis	[P5]

30	SACAMCS	Software Architecture Comparison Analysis Method for Critical Systems	[P61]
31	Taguchi	Taguchi technique	[P32]
32	UAV	Utility Value Analysis	[P71]
33	VIKOR	Multi-criteria Optimization and Compromise Solution	[P25]

After the analysis of those techniques, some interesting phenomena have been identified (Figure 9). For instance, AHP had the highest frequency of use in the process of software development, which were about 15 publications of 56 primary studies applied it exclusively or used a hybrid methodologies which AHP was also involved. The second most used technique was Pareto Optimization, which was designed to resist modelling and measurement errors present in multi-objective decision problem under uncertainty (Emmanuel, David, & Earl, 2014), and about nine publications were related to it. Besides AHP and Pareto Optimality method, TOPSIS, WSM (Weighted Sum Model), MAUT (Multi-Attribute Utility Theory), OWA (Ordered Weighted Averaging / Aggregation), QFD (Quality Function Deployment), and Outranking were also popular in the development of software.

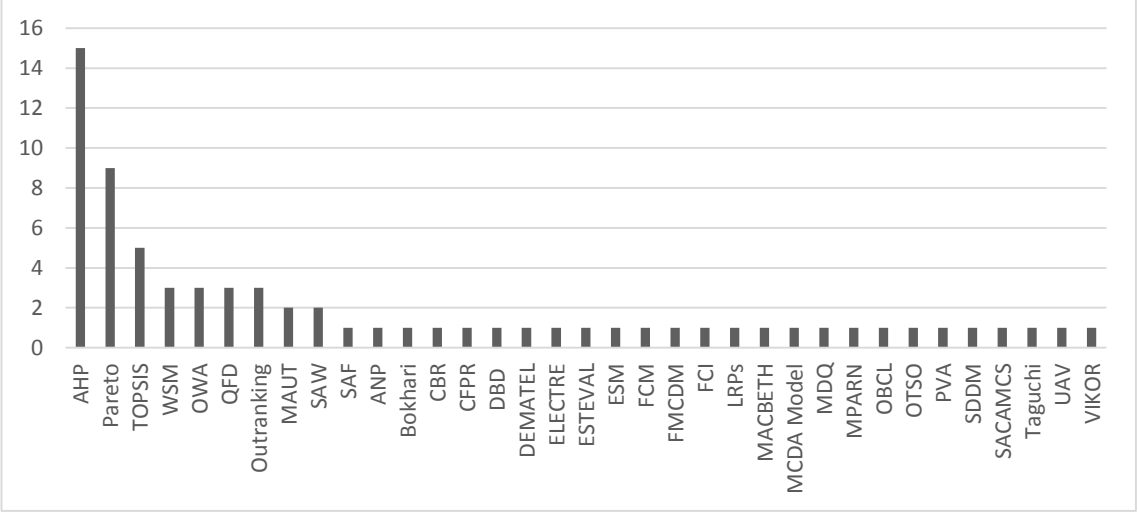

Figure 9. The frequency of MCDM techniques in software development.

Analytic Hierarchy Process

Analytical Hierarchy Process (Saaty, 1980) is a popular MCDM method which helps decision makers to resolve the multi-criteria problems by decomposing them into a hierarchical model (Norita & Phillip, 2006; Ying-Fu & Ming-Hui, 2010). Generally, the hierarchical model is comprised by three layers. The top layer represents the goal developers want to achieve after the process of decision making, the second layer is the criteria which used for evaluating the alternatives, the lowest layer represents the alternatives which are the candidates of the objectives of decision making (Saaty, 1980). The key aspect of AHP is the transition between the second layer and the third layer. In other words, the difficult part of AHP is how to evaluate those alternatives, in most of the cases, pairwise comparisons are applied (Saaty, 1980).

Pairwise comparisons are conducted according to a certain scale (from 1 (equal importance) to 9 (extreme importance of a criterion compared with the other criterion in the pair)) to determine which of the two criteria being compared is more important and how much more important it is (Alexander & Emmanouil, 2013; Omolade & Guenther,

2005). After pairwise comparisons, a comparison matrix must be created for each criterion (Marina, Jocelyn, & Hernán, 2014). A set of weights which represent the grades of preference between criteria would be generated by the process of creating the comparison matrix (Marina, Jocelyn, & Hernán, 2014). When the comparison matrix is ready, the eigenvector of the relative importance of the criteria needs to be computed, it is then used to rank the alternatives (Khan, Azra, & Mohd, 2014). In the end, the optimal option can be selected based on the ranking.

Table 17. AHP and corresponding MCDM problems.

MCDM Method	MCDM Problems
AHP (Saaty, 1980)	[P15] [P74] [P39]Components selection (P74: Fuzzy AHP)
	[P62] Design Optimization
	[P37][P48] Selection of software development life cycle model (P48: Fuzzy AHP)
	[P51] Selection of software project management tool
	[P41] Selection of tools
	[P43] Resource Allocation
	[P53] Measurement of Component Modifiability
	[P3] Evaluation of quality trend
	[P24] [P70] Selection of system alternatives
	[P71] Evaluation and analysis of alternative systems
	[P60] Requirements tradeoff analysis

Fifteen primary studies used AHP as the means to make decisions (Table 17). 10 out 15 primary studies applied AHP technique for the selection of components or tools or concepts, such as selection of components, selection of software development life cycle models, selection of tools, and selection of systems (P15, P74, P37, P48, P51, P41, P24, P70, P39 and P71). Therefore their hierarchical models are basically the same, the top level represents the objectives of selection, the second level represents the quality attributes of the things which would be selected, such as performance, security, response time. The lowest level represents the candidates, for example, the candidates of software development life cycle models can be waterfall model, spiral model, incremental model (P37, P48).

Sometimes the AHP criteria are hard to precisely define or describe, so it is not easy for decision makers to make correct judgment between criteria (Ying-Fu & Ming-Hui, 2010). To overcome the shortages of AHP and to resolve the vagueness of AHP criteria, a hybrid method which combines the benefits of both fuzzy set theory and AHP was proposed (Laarhoven & Pedrycz, 1983). Two out of 15 primary studies applied fuzzy AHP (P48, and P74) to help decision makers to define those vague criteria. Fuzzy set theory is a mathematical theory which is designed to model the concepts of vagueness, imprecision and uncertainty in human cognitive processes (Zadeh, 1965), in other words, fuzzy set theory can assist developers to explain vague, imprecise and uncertainty concepts within human cognitive; therefore, the differences between fuzzy AHP and AHP are that fuzzy AHP can handle the vague imprecise and uncertainty criteria.

Pareto Optimization

As mentioned in chapter 2, multi-criteria decision making has been divided into multi-attributes decision making and multi-objectives decision making. Multi-objectives

decision making methods are always applied for the design and planning phases, which aims to achieve the desired goal by considering infinite number of continuous alternatives (Mateo, 2012; Gwo-Hshiung & Jih-Jeng, 2011). While Pareto optimization is also called multi-objectives optimization used to resolve two or more incomparable and conflicting objectives (Miettinen, 1998; Hwang & Masud, 1979). For the Pareto optimization problems, a single solution which can simultaneously optimize all those objectives does not exist. In that case, a Pareto optimal which is a set of candidate solutions will be created (P2, P7, P14, P19, P20, P31 and P69). In the Pareto optimal set, each candidate solution is equally important, in other words, they are not dominated by any other (Colin, Cristinel, & Romeo, 2008).

Pareto optimal set is also called Pareto frontier, which describes the tradeoffs between objectives, so the best solution can be selected (Fatah, Kash, & Andrea, 2012). Nevertheless, there are still difficulties of using Pareto frontier for the best design selection which are computing the Pareto frontier for multi-objective problems, determining the set of solutions representing the frontier, and performing a sensitivity analysis of the frontier (Fatah, Kash, & Andrea, 2012). Even though there are some difficulties of using Pareto frontier, it can still help decision makers to identify what can be achieved and to select one candidate solution in that set that corresponds to an appropriate trade-off among the multiple objectives (William & Emmanuel, 2011).

Table 18. Pareto optimization and corresponding MCDM problems.

MCDM Method	MCDM Problems
Pareto optimization (Miettinen, 1998; Hwang & Masud, 1979)	[P2][P7] Selection of system design concepts
	[P69] Design Optimization
	[P2][P14] Design space exploration
	[P69] Simulation of system design
	[P31] Customization of software engineering technologies
	[P20] Candidate layouts and operational designs sensitivity analysis
	[P19] Evaluation of early requirements
	[P72] Performance Evaluation
	[P18] Selection of control laws

There are amounts of methods and different kinds of processes utilized to conduct the Pareto optimization (P2, P7, P14, P19, P20, P31, P72, P18 and P69) (Table 18), ten primary studies used it to resolve their multi-objectives decision making problems in our literature review (Table 6). Particularly worth mentioning was that eight of them are related to design (P2, P7, P14, P20, P31, P72, P18 and P69).

Technique for Order Preference by Similarity to Ideal Solution

The technique for order preference by similarity to ideal solution (TOPSIS) is another well-known MCDM technique in which the chosen alternative should have the shortest geometric distance from the positive ideal solution and the longest geometric distance from the negative ideal solution (Hwang & Yoon, 1981). Similar with AHP, TOPSIS also considers multiple alternatives by evaluating multiple criteria, but what is different is that TOPSIS uses matrix to do the evaluation. The first step of the decision making process is creating the evaluation matrix which consists of multiple alternatives and criteria. In order to allow a comparable scale for all criteria assessed by different measurement units

(Ullah, De-Qun, Peng, Mukarrum, & Amjad, 2013; Jiunn-Chenn, Taho, & Cheng-Yi, 2010), the normalization of the matrix should be done after the first step.

There are two methods in order to normalize the matrix, they are linear normalization and vector normalization (Hwang & Yoon, 1981). To find the normalized value, the vector normalization approach divides the rating of each attribute by its norm, while the linear normalization approach divides the ratings of a certain attribute by its maximum value (Jiunn-Chenn, Taho, & Cheng-Yi, 2010). After the normalization of decision matrix, the calculation of the weights of the normalized matrix should be ready because the positive ideal solution (the best alternative) and the negative ideal solution (the worst alternative) are needed to be determined. The geometric distance between the targeted alternatives and the best and worst alternative can then be calculated. Based on the distances, the similarity to the worst alternative can be measured. The last step of TOPSIS is ranking the alternatives according to the similarity (Hwang & Yoon, 1981). TOPSIS is easy to understand, and because it is a kind of compensatory method, so it allows the tradeoffs between attributes (Mumin, 2012).

Table 19. Technique for Order Preference by Similarity to Ideal Solution and corresponding MCDM problems.

MCDM Method	MCDM Problems
Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) (Hwang & Yoon, 1981; Yoon, 1987)	[P72] Performance Evaluation
	[P73] Selection of data mining algorithms
	[P48] Selection of software development life cycle model
	[P32] Selection of optimal scenarios
	[P67] Selection of system design concepts

Five primary studies used TOPSIS for their decision makings (P72, P73, P48, P32, and P67) (Table 19), and four of them were about the selection (P73, P48, P32, and P67), like selection of data mining algorithms, selection of software development life cycle model, selection of optimal scenarios, and selection of system design concepts.

Weighted Sum Model

Similar with TOPSIS, weighted sum model also starts with a decision matrix which consists of multiple alternatives and criteria. While the data expressed in the matrix of weighted sum model (WSM) are in the same unit, it means it does not need the normalization process like TOPSIS, i.e., the process of decision making is greatly simplified; therefore, weighted sum model is the simplest MCDM method (Fishburn, 1967). Decision makers simply need to calculate the WSM scores by considering the weights of its criteria and alternatives, WSM scores are then ranked according to their weights (Fishburn, 1967). Table 20 shows the MCDM problems which resolved by weighted sum model from the primary studies. Three primary studies have used this method to make decision (P21, P18, and P66).

Table 20. Weighted Sum Model and corresponding MCDM problems.

MCDM Method	MCDM Problems
Weighted Sum Model (Fishburn, 1967)	[P21] Component Selection
	[P18] Selection of control laws
	[P66] Requirements tradeoff analysis

Ordered Weighted Averaging

The OWA operator is usually used to reorder fuzzy numbers in a fuzzy environment, so the reordering process is a fundamental aspect of it which associates the arguments with the weights (Lamata, 2004). An OWA operator of dimension n is a mapping $F: R_n \rightarrow R$ that has an associated n vector

$$w = (w_1, w_2, \dots, w_n)^T \quad (2)$$

Such as $w_i \in [0,1]$, $1 \leq i \leq n$, and

$$\sum_{i=1}^{n} w_i = w_1 + \cdots + w_n = 1. \quad (3)$$

Furthermore

$$F(a_1, \dots, a_n) = \sum_{j=1}^{n} W_j B_j = W_1 B_1 + \cdots + W_n B_n \quad (4)$$

Where b_j is the jth largest element of the collection$< a_1, \dots, a_n >$.

Table 21. Ordered Weighted Averaging and corresponding MCDM problems.

MCDM Method	MCDM Problems
Ordered Weighted Averaging (OWA) (Yager, Families of OWA operators, 1993; Yager, Constrained OWA aggregation, 1996)	[P56] Model evaluation, comparison and selection
	[P35] Selection of configuration items
	[P26] Risk assessment

In our study, there were three primary studies mentioned OWA operators and used them in their process of decision making (Table 21). OWA operator was utilized to evaluate, compare and select model in primary study 56. It mentioned that the OWA operator with the fuzzy linguistic quantifier "most" was used to aggregate the preference matrices across all criteria, thus the aggregation provided a preference matrix bearing in mind "most" of the criteria (Rajabally & Holywell, 2006). In primary study 35, the OWA operator was applied to aggregate the individual dominance degrees of a candidate item for all experts, thus it resulted in a completed order of candidate items at last (Juite & Yung-I, 2003). In primary study 26, OWA operator model was utilized to evaluate risk in software development. During the whole process of decision making, OWA aggregation model was used in step 4 and step 6 to get the refined weights of attributes for the evaluation (Hing-Hsue, Jing-Rong, & Tien-Hwa, 2006).

Quality Function Deployment

QFD is a method which can transform qualitative user demands into quantitative parameters, deploy the functions for forming quality, deploy methods for achieving the design quality into subsystems and component parts, and ultimately to specific elements

of the manufacturing process (Akao, 1994). Some techniques and tools which are based on QFD can be used to conduct multi-criteria decision making, for example, Pugh concept selection which is invented by Stuart Pugh can be used to select an optimal item among a list of candidates by in coordination with QFD. Besides it, House of Quality and Modular Function Deployment are also very useful tools which are also deprived from QFD (Akao, 1994).

Table 22. Quality Function Deployment and corresponding MCDM problems.

MCDM Method	MCDM Problems
Quality Function Deployment (QFD) (Akao, 1994)	[P65] Selection of system design concepts
	[P16] Cost Estimation
	[P47] Performance Evaluation

Table 22 shows that three primary studies utilized QFD to make decisions when they need to cope with multiple criteria. In primary study 65, a simple House of Quality matrix was used to quantify the relative importance of each criterion of each requirement, the design team could then make a selection of the design features (Thomas & Christian, 2007). QFD methodology was integrated with a cost estimation model to assist decision making in software designing and development processes for improving the quality (Divya & Misra, 2013). The planning matrices of QFD was also used to deploy requirements by the degree of importance of these requirements, then started to measure performance of the manufacturing system (Muhamad, 1997).

Outranking

Outranking methods were firstly mentioned in scientific paper by Roy Bernard in the late 1960s (José, Salvatore, & Matthias, 2005), since then, increasingly attentions were paid to outranking models, especially in Europe. Three classes of outranking methods are existed nowadays: PROMETHEE (Brans & Vincke, 1985), ELECTRE (José, Salvatore, & Matthias, 2005), ORESTE methods (Roubens, 1982). The basic ideas of outranking methods are that a preference relation also called outranking relation is built among alternatives evaluated on several attributes, and in most of outranking methods, the outranking relation is built through a series of pairwise comparisons of the alternatives (José, Salvatore, & Matthias, 2005). The pairwise comparison reminds us the similar term in AHP, the process of this pairwise comparison is different with the one in AHP, but the concordance-discordance principle between them is the same.

Assume that there is a set of alternatives and a set of criteria, alternatives will then be denoted by symbols $a, b, c, \ldots$, criteria are denoted by $Gj, j = 1, \ldots, n$, in the end, $Gj(a)$ represents the score of alternative a on the jth criterion (José, Salvatore, & Matthias, 2005), and this is also the basic idea of ELECTRE (Elimination and Choice Translating Reality, translated from French). Another outranking method called Preference Ranking Organization Method for Enrichment Evaluations (PROMETHEE) is a well-established decision support system which copes with the appraisal and selection of a set of alternatives on the basis of several criteria, and the objective of this MCDM method is obtaining a ranking among them by identifying the pros and the cons of the alternatives (Brans & Vincke, 1985). The processes of PROMETHEE can be concluded as two steps, first it should assign a preference function, and then estimate the outranking degree of the alternatives (Brans & Vincke, 1985). Different kinds of PROMETHEE tools or modules have been developed, like PROMETHEE I, PROMETHEE II and GAIA, and they are used for different objectives, for example, PROMETHEE I is utilized for partial ranking,

PROMETHEE II is used for complete ranking and GAIA is applied for visualization (Geldermann & Rentz, 2001; Macharis, Springael, De Brucker, & Verbeke, 2004).

Table 23. Quality Function Deployment and corresponding MCDM problems.

MCDM Method	MCDM Problems
Outranking (PROMETHEE II (Brans & Vincke, 1985) and ELECTRE (José, Salvatore, & Matthias, 2005))	[P31] Customization of software engineering technologies
	[P73] Selection of data mining algorithms
	[P24] Selection of system alternatives

In this study, two primary studies applied PROMETHEE II in their researches (P73, P24) to do the selection and one primary study used ELECTRE-IS to customize software engineering technologies (P31).

5. Discussion

This chapter aims to discuss the results which have been presented in Chapter 4. Similar with the previous chapter, this chapter consists of three sub-chapters which are based on three research questions, there would be several aspects which are the implications observed from the results below each sub-chapter.

5.1 Software Development Problems (RQ1)

About 33 types of software development problems were identified after the process of SLR, and six categories were classified based on their characteristics, the problems in each category have some features in common, for example, the category of software components encompassed all the problems which were related to software components, whether they were about the evaluation of components or the selection of components. Two aspects of software development problems were detected by our observation after the data extraction, one was that there are three the most frequent occurred problems which were all involved with five primary studies, and the other one was that a specific MCDM problem could be resolved by different MCDM methods, for example, the problem of performance evaluation could be addressed by five different MCDM methods. In this section, these two interesting findings will be discussed and analyzed and a conclusion will also be reached to answer the first predefined research question.

5.1.1 The Most Frequent Occurred Problems

Three MCDM problems were identified as the most frequent occurred problems from 33 different problem types. They are components selection, design concepts selection, and performance evaluation. This section will analyze all of these problems based on the relevant primary studies, and in the end, summarize their common characteristics and discuss their differences.

Components Selection

An individual software component is a software package, a web service, a web resource or a module which encapsulates a set of functions or data, and it can communicate with other components via interfaces (Niekamp, 2005). The selection of components is one of the most frequent occurred problems which needs to be resolved in the process of MCDM according to our study. There are five primary studies relating to components selection problems (P15, P29, P39, P58, and P74). Cooper et al. (2007) stated that each component had its own QoS (quality of service) behaviors, for example, memory usage, response time, and a replacement component needed to be selected with regard to the current QoS objectives in the system. The reasons of selecting replacement components were that the objectives of current QoS lead to the search for a component with the same capabilities, but used a low or moderate amount of memory (P15). According to Cooper et al. (2007), the selection of appropriate components was conducted based on the status of the system and the relative importance of selected features (P15).

Jett and Midkiff (1995) thought that the software designers must consider alternate implementation of the functional structure of system. That is to say, the replacement of system's functional structure, and select them among various possible allocations of functions to hardware, software, and firmware which were referred to a selection of particular hardware parts or software modules, which were also named component alternatives (P29). Each alternative has its attributes, such as cost, performance,

reliability, so the designers must evaluate those attributes for each alternative and guide the process of MCDM (P29). Künzli et al. (2005) demonstrated the COTS selection for the development of COTS systems in their study, the reasons for COTS components selection are that COTS system need to be customized with regard to individual requirements, as a result, it is critical for subsequent phases of the software development life cycle to select the proper COTS components which involve multiple objectives, such as cost, compatibility, the ease of installation (P39).

Rose et al. (2005) stated that the aims of components selection in their article were leading to an optimized design which means better component matching, increased efficiency, lower cost, ease of system integration, compatibility with other systems, improved controllability, increased reliability and other non-functional requirements, by which complex engineering decision making under multiple design criteria could be simplified into a few simple and straightforward decisions (P58). Ying-Fu and Ming-Hui (2010) stated that the purpose of their study was to obtain an overall view of system by identifying the relative importance of design features which can be applied as a design strategy by developers making tradeoffs between the design features during the product development process (P74). This study seemed not to be related to components selections, but it was about identifying the relative importance of design features which can also be called components, what it missed was the part of ranking those components by considering their relative importance.

After analyzing those primary studies related to components selection, a summary about the characteristics of this MCDM problem and the motivation on conducting the selection of components were obtained. The basic common characteristics of components selection in those primary studies were that they all related to the multiple quality attributes of component, such as performance, reliability, cost, and all those attributes needed to be concurrently resolved. However, what different were their purposes when they were selecting components. For example, some studies were selecting the components for replacement in order to achieve higher QoS, the aims of some studies are simply to select appropriate components for an optimized design in their project development, and the purposes of some studies are gaining an overall view of system by evaluating all the components.

Design Concepts Selection

The design concepts provide the software designer with a foundation from which more sophisticated methods can be applied (Suryanarayana, Samarthyam, & Sharma, 2014), so issues of design concepts selection are crucial for software designers in the phase of system analysis and design, and the selection of design concepts are also one of the most frequent occurred problems during the software development process (P2, P7, P54, P65, and P67). Alessandro et al. (2006) stated that concept selection is important, because poor selection of a design concept can rarely be compensated for at later design stages and results in great redesign costs; therefore, the major task of system level design is to generate design concepts, evaluate them, and choose one or more best concepts for further refinement in the latter design stages (P2). Aniela et al. (2003) demonstrated that their tenet is exploring the design space thoroughly and computationally during conceptual design which the designer can select the most promising system concepts in the best position (P7).

Pierre et al. (2014) stated that software engineers require supports in reviewing alternative system design solutions, and in making and defending the best design choices in the earliest product design stages (P54); therefore, a rank of design solutions is necessary for

them. Thomas and Christian (2007) said that different design concepts for systems were derived by one MCDM method, and they were assessed based on specific criteria linked with requirements in their study (P65). Ullah et al. (2013) stated that software engineers who are involved in generating new design concepts not only need to consider the required technical performance but also need to consider cost, manufacturability, reliability, environmental friendliness and quality of the end product. The objective of this study was to enrich the existing concept generation approaches and thus be able to select the best design concept at first time and avoid the costly design changes at a later stage (P67).

After conducting the summarization from all these five primary studies which were related to the design concepts selection, it was realized that the common characteristics of this MCDM problem were that this problem usually happened in the earliest system design stages, and poor performance of design concepts selection would result in great costs in latter design stages. While their objectives were different when they were conducting the concepts selection, some studies aimed to have a better further refinement in the latter design stages, some studies aimed to explore design space for the efficiency and convenience of designers' selection of system design concepts, while some studies intended to avoid costly design changes in latter stage by selecting the best design concepts.

Performance Evaluation

Software Performance represents the entire collection of software engineering activities which encompasses the set of roles, skills, activities, practices, tools, and deliverables and related analyses used throughout every phase of the software development life cycle, which are directed to meet non-functional requirement for performance, such as reliability, security, maintainability, latency, controllability (Woodside, Carleton, Franks, & Petriu, 2007). System performance evaluation has become one of the most occurred MCDM problems during the process of software development based in our study (P25, P5, P47, P55, and P72). Gülçin and Da (2008) set forth that the purpose of their study was measuring performance of software development project. The reasons of great importance of software product measurement were concluded in two reasons, one was that the demand for qualitative and reliable software which complied with the international standards and was easy to integrate into existing system structures was constantly increasing, and the other one was that the cost of software production and maintenance was dramatically growing. As a result, the complexities and the needs for better designed and user-friendly software products were also increased (P25).

Andrea et al. (1996) stated that the measurement of performance would be useful during the design phase which possessed properties congruent with the objectives of the problem, and it should relate to the strategic as well as operational objectives of the system (P5). Muhamad (1997) elaborated that design parameters such as throughput time, work-in-progress, manufacturing cost, product quality, can affect the operational performance of manufacturing system, so the performance of system can be measured along those parameters (P47). Raffo et al. (2002) elaborated how to accomplish tradeoffs among performance measures in their study, and a performance picture of the project was achieved to manage those tradeoffs (P55). Xiaoqian and Yongchang (2010) explained that the evaluation of the control system performance consisted of the tradeoff between multiple, potential conflicting criteria; consequently, considering the multiple conflicting performance criteria simultaneously needed to be prudently conducted (P72).

After analyzing all those five primary studies related to performance evaluation, the common features of this MCDM problems were concluded, which were that all the performance criteria are conflicting and considering them simultaneously needs great prudent efforts. What different was that those five primary studies were about five different types of systems, which were ERP system (P25), flexible manufacturing system (P5), manufacturing system (P47), and control system (P72), and the methods they used to cope with the evaluation were also different.

5.1.2 Different Methods Solve One Problem

The three most frequent occurred MCDM problems have been analyzed which were components selection, design concepts selection and performance evaluation. Some interesting phenomena have been identified that even though some primary studies had the same development problems, but the methods they applied were different. Based on those phenomena, two questions were raised that why their MCDM methods were different, and whether there were any commonalities among those methods. This phenomena will be discussed and the questions will be answered based on the analysis of components selection and its corresponding methods. Four different MCDM methods have been identified to cope with the problem of components selection, they were AHP (P15, P74), ESTEVAL (P29), OTSO (P39), and MDQ (P58). The conclusions about the common characteristics of components selection have been reached, which were about resolving multiple attributes simultaneously.

For AHP, this problem will be straightforwardly decomposed into three levels, the top layer represents the highest level objective, the second level represents the attributes, such as memory, response time, security, data integrity, the lowest level represents the alternative components that are available (Norita & Phillip, 2006; Ying-Fu & Ming-Hui, 2010), and the problem will then be solved based on this hierarchical model. For ESTEVAL, a unique set of values for metrics such as cost, performance, and reliability, will be possessed by each alternative, and the designers will estimate the metric for each alternative and use the metric values to compare the alternatives. As a consequence, ESTEVAL addressed the problem by maintaining values for multiple metrics and providing displays to help the designer recognize problem areas and explore the design space (Jett & Midkiff, 1995). What important was that it embodied the concept of concurrent engineering which emphasized the simultaneous considerations of multiple design attributes (Jett & Midkiff, 1995).

With regard to OTSO, the first step was defining criteria which are classified into four groups which are functional criteria, quality criteria, strategic criteria, and domain and architecture criteria; the next step was identifying, filtering and evaluating components; the last step was analyzing those components by applying AHP method, which used pairwise comparisons to determine the importance of criteria on each level, then checked the consistency of rankings, and finally, evaluated the candidate components and presents the recommendation (Künzli, Lothar, & Eckart, 2005). With regard to MDQ, the complex engineering decision making under multiple design criteria were simplified into a few simple and straightforward decisions, and finally leaded to an optimized design (Rose, Clarence, Marcelo, Jim, & Henk, 2005).

After analyzing the problem of component selection and its different methods, the questions which raised in the first paragraph of this section were answered. The reasons of why their methods were different were explained from the previous analysis of all those five primary studies, which had different contexts of problems. Their motivations of doing the component selection were also different, in other words, what they want to

obtain after the decision making were different, so the methods they chose to apply were different. The next question was that whether there were any commonalities among those methods. AHP, ESTEVAL, and OTSO had commonalities which were their ways to compare alternatives, they all assigned weights to their alternatives, and then compared those weights. But what different were that ESTEVAL assigned weight in a different way, it assigned each alternative a unique set of the values of metric, AHP and OTSO assigned their weight by pairwise comparison. MDQ was total different with others, because it simplified the complex decision problems into some simple and straightforward decisions. What can be learned from the case of component selection were that there were amounts of MCDM methods can be applied to solve the decision making problems, those methods were selected based on the contexts of their corresponding problems, and the strengths of other methods can be applied in future decision making processes.

5.2 Software Development Process Phases (RQ2)

All those 56 primary studies were grouped into six categories based on the software development process phases of their MCDM problems. Some interesting phenomenon about the distribution of those primary studies were found after the categorization, the phase of analysis and design owned the largest number of primary studies, and system build/prototype/pilot phase and implementation and training phase had zero primary studies involved.

5.2.1 MCDM Problems in Software Design phase

A software design is a description of the software structure, the data which is part of the system, the interfaces between system components and the algorithm used (Sommerville, 2004); therefore, the problems occurred in this phase are always related to analysis and design, such as, design optimization, design concepts selection, components selection, system design simulation, evaluation and analysis alternative system. These problems are all important for the software development, for example, component selection can help software developers achieve an optimized design or gain an overall view of system or achieve a higher quality of service, design concepts selection can promote a better further refinement in the latter design stages or avoid costly design changes in latter stage by selecting the best design concepts.

Imagine that if these problems were left to next phases, as a result, how the software perform and function could not be better understood, and what was more, the way it should perform and work could not be clearly expressed; what the requirements for functionality are could not be defined and understood as well; and the plans and budgets to choose the best way of implementation could not be considered (Jacobson, Booch, & Rumbaugh, 1999). While MCDM is a formal approach to assist decision makers to reach a compromise or consensus among criteria which are relatively precise but generally conflicting (Belton & Stewart, Multiple Criteria Decision Analysis: an integrated approach, 2002; Stewart, 1992). Multiple objectives and criteria need to be considered simultaneously when a concept or component or a tool is selected in the software design phase, so most of the problems which occurred in this phase are MCDM problems (Table 13); therefore, it was explained that why MCDM problems occurred most in analysis and design phase, it was because all the activities occurred in this phase were about making better choices from multiple alternatives with multiple criteria, and these activities could not be left to next phase especially when a better received vision of the software product was needed.

5.2.2 MCDM Problems in Implementation and Validation phase

The main works of software implementation are basically programming and debugging. All these activities are carried on by following the instructions which are made during the previous phases, such as feasibility study, requirement definition, analysis and design. Software validation involves testing processes to inspect and review whether the system complies with its specification and meets users' expectations (Sommerville, 2004). With regard to software product, the main work in this phase is building, distributing, installing, configuring, testing, and executing systems that move through the software development process. While decision making activities are more likely to analyze, evaluate and decide the issues occurred in the early phases or sustainment phase. As a result, it is not difficult to explain why there is no MCDM problems in this phase.

5.3 MCDM techniques (RQ3)

33 MCDM techniques have been extracted from 56 primary studies, each technique has its own characteristic and they all can be applied to make multi-criteria decisions in software development. Many interesting phenomena were observed from the extracted results, for example, AHP is the most frequent used MCDM method during the development of software, some studies don't only use one MCDM method and they prefer to combine them into hybrid methodologies to cope with the MCDM problems, different contexts of decision making needs different techniques. Those aspects will be discussed and explained based on the primary studies.

5.3.1 The Most Frequent Used MCDM Method

Fourteen primary studies have used AHP to resolve their MCDM problems (Table 5), which was 26.8 percent of all the primary studies. The reasons of AHP is the most popular MCDM techniques for software engineers to make decisions during the software development can be identified from the primary studies. Alexander and Emmanouil (2013) emphasized that the main strengths of AHP were that AHP was able to decompose a decision problem into its constituent parts which stressed the importance of each criterion. The capability of inconsistencies checking and the convenience of AHP were also the advantages of it (P3). Khan et al. (2014) used the process of AHP to resolve the complex problem with multiple conflicting and subjective criteria by permitting the hierarchical structure of the criteria or sub-criteria when allocating a weight (P37). Norita and Phillip (2006) complemented AHP on the way it incorporated multiple experts' opinions and control of consistency in judgments. In addition, the guarantee of high repeatability and scalability controls were also one of strengths of AHP (P51).

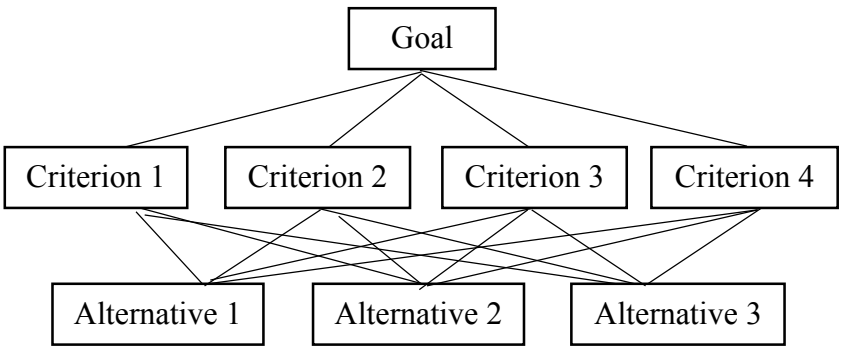

Figure 10. A simple hierarchy of AHP (Saaty, 1980).

Mindy et al. (1999) also commented that AHP can organize the basic rationality of the priority setting process by breaking down a multi-elements complex system into its smaller constituent parts (P43). Most of the MCDM problems were about the selections and evaluations during the development of software (Table 5 – Table 10), and most of them are related to multiple criteria. Since AHP has three layers which consists of goal, criteria and alternatives, it quite liked the way human think about the decision making, so it is quite clear when people makes decisions (Figure 10). The key step of AHP is the pairwise comparison which compare two criteria based on a certain scale which from 1 (equal importance) to 9 (extreme importance of a criterion compared to the other criterion in the pair), so it is simple and easy to use, and it allows decision makers to determine the tradeoffs among criteria (Alexander & Emmanouil, 2013; Omolade & Guenther, 2005). Most importantly, no matter what types of criteria were evaluated, quantitative or qualitative, the weights can always be assigned to them.

In sum, reasons have been summarized about why software engineers like to use AHP for their decision making. First, it is similar with human's thinking about decision making; second, it is simple and easy to use; third, it can decompose complex problem into small parts; fourth, it allows decision makers to determine the tradeoffs among criteria; last but not the least, all kinds of criteria can be used, no matter those criteria are qualitative or quantitative. As a conclusion, AHP is a good and reliable MCDM method.

5.3.2 Hybrid Methodologies for MCDM

Sometimes only one single MCDM method is not good enough to deal with the multi-criteria decision problems, it needs hybrid methodology which takes the strengths of each MCDM method and combines them together to make a better decision. Hybrid methodology of multi-criteria decision making in software development is a quite common approach to help software engineers to make decisions (P41, P48, P24, P71, P40, P32, P2, P31 and P18). In our 56 primary studies, 9 primary studies applied hybrid methodologies in their decision making, and there were three main categories of these hybrid methodologies, one was combining with AHP which means AHP was always one part of the decision making process, one was combining with TOPSIS, the last one was about combining with Pareto optimization. Three categories will be discussed and the common characteristics of AHP, TOPSIS and Pareto optimization will also be concluded.

Combine with AHP

AHP is one of the most known MCDM methodologies (Saaty, 1980), combing with AHP as a hybrid methodology is also popular manipulation in the field of MCDM (P41, P48, P24, and P71). After the process of SLR, it has been found that four primary studies have applied AHP, meanwhile with other MCDM methods, they were AHP with MAUT, AHP with TOPSIS, AHP with PROMETHEE, and the combination of UAV, AHP and ANP. In the combination of AHP and MAUT, MAUT was used to complement AHP which AHP was used to obtain the weights indicating the selection criteria preference scale, and MAUT was used to evaluate the alternatives (P41). The reasons of using a combination of AHP and MAUT in primary study 41 were that it can reduce the amount of information that had to be entered by development team, because it is relatively easy to assign weights by using AHP, since only the superior triangle of the criteria comparison matrix needed to be filled by user, it required less user inputs than that required values for all pairwise comparisons of AHP (P41).

The combination of AHP and TOPSIS was conducted in primary study 48. The whole decision process was divided into three phases, the first phase was about determining

alternatives and criteria which would be used for the evaluation of alternatives, and constructing the decision hierarchy by using the determined alternatives and criteria. In the second phase, AHP was used for assigning the criteria and sub-criteria weights. In the last phase, TOPSIS was used to determine alternatives' priority weights (Mumin, 2012). In primary study 24, the AHP method was applied to derive and assign weights for each criterion. The pairwise comparison took from AHP was used to evaluate the comparison of each item by experts, while PROMETHEE was applied to provide a complete order for the evaluation that will help decision makers easily realize the evaluation results (Gwo-Hshiung, Tzay-An, & Chien-Yuan, 1992). The last hybrid methodology was a rare combination of three MCDM methods which are UAV (utility value analysis), AHP and ANP (analytic network process) (P71). In the first stage, UAV was used to obtain the first list and gain the overview of alternatives; AHP was applied to formalize the rating and evaluation process, thus increasing the decision quality in the second stage; in the last stage, ANP was utilized to allow the alternative-specific evaluation criteria which furthermore improves the precision of the decision process (Vinzent, Jürgen, & Gerald, 2014).

From the observations of the decision making process of these four primary studies, it was clearly realized that AHP is always used to assign the criteria weights. Reviewing back to the description of AHP, it was found that the process of assigning criteria weights is based on pairwise comparison, pairwise comparisons are conducted according to a certain scale to determine which of the two criteria being compared is more important and how much more important it is (Alexander & Emmanouil, 2013; Omolade & Guenther, 2005). And no matter what types the criteria are, decision makers can assign weights to them according to their judgements, especially when the criteria are fuzzy. Consequently, the common characteristics of those hybrid methodologies were that AHP was always applied to assign the criteria weights because of its pairwise comparison.

Combine with TOPSIS

TOPSIS is another well-known MCDM method, which is based on the concept that the chosen alternative should have the shortest distance from the positive ideal solution and the longest distance from the negative ideal solution (Hwang & Yoon, 1981). Combing with TOPSIS as a hybrid methodology was also occurred during the decision making process in software development (P32, P48). Before introducing the combination of Taguchi and TOPSIS, a brief description of Taguchi methods will be presented. Taguchi methods are statistical methods developed by Genichi Taguchi to improve the quality of manufactured goods, which includes three principal contributions to statistics, a specific loss function, the philosophy of off-line quality control and innovations in the design of experiments (Taguchi, 1993), and the Taguchi method is applied on the aspect of a specific loss function in primary study 32.

The main essence of hybrid Taguchi and TOPSIS was the notion of quality loss transformation which was about transforming the performance measures into quality loss function, in order to conduct the transformation, a Taguchi experimental design was planned, as a result, the decision matrix was generated after the Taguchi experiment, then the decision matrix was one part of the TOPSIS decision making process (Jiunn-Chenn, Taho, & Cheng-Yi, 2010).While in primary study 48, two different MCDM methods were carried on different phases, they were complementary for each other. AHP was applied in the second phase to assign weights to criteria and sub-criteria, while TOPSIS was used to determine alternatives' priority weights in the third phase. The connections between the second phase and the third phase were the decision matrix which was generated by AHP in phase 2. The decision matrix was then used in phase 3 through the process of

TOPSIS (Mumin, 2012). As a result, it was concluded that the common characteristics between the two cases of combing with TOPSIS were that the only place which needed to be optimized in TOPSIS process was the generation of decision matrix, after obtaining the decision matrix, the routine of decision making was still carried by following the instruction of TOPSIS.

Combine with Pareto Optimization

Pareto optimization also called multi-objectives optimization can be used to resolve two or more incomparable and conflicting objectives (Miettinen, 1998; Hwang & Masud, 1979), while a single solution which can simultaneously optimize all those objectives does not exist; therefore a Pareto optimal which is a set of candidate solutions should be created (P2, P7, P14, P19, P20, P31 and P69). Three hybrid methodologies related to Pareto optimization were applied among the 56 selected primary studies, they were FCM (Fuzzy Clustering Means) with Pareto optimization (P2), Outranking (ELECTRE-IS) with Pareto optimization (P31), and WSM (Weighted Sum Model) with Pareto optimization (P18). The details of them will be discussed. The combination between FCM and Pareto optimization was utilized to select design concept and explore design space. In this study, Pareto optimal set was a set of candidate design solutions generated by the strategy of multi-objective design space exploration, the Pareto set was then equally divided into an optimum number of clusters which was useful to determine a more balanced Pareto subset by using a clustering algorithm (Alessandro, Maurizio, & Davide, 2006).

The second hybrid methodology was the combination between ELECTRE-IS and Pareto optimization. In this study, the main reason of applying the Pareto optimization approach was that more data points for each heuristic over a data set can be used for searching the optimal solutions, consequently, it was likely to consider more candidate alternatives (Jingzhou & Guenther, 2008). However, they also selected ELECTRE-IS, the rationale of their choice of ELECTRE-IS was based on its ability to handle uncertainties in preference building. The final outranking relation of ELECTRE-IS expressed by a directed graph helped decision makers to decide which heuristic was preferable to others in which situation (Jingzhou & Guenther, 2008). As a result, Pareto optimization was used to recommend heuristic by considering the relative importance of the criteria, while ELECTRE-IS was applied to select heuristic based on partial ordering obtained from outranking graph. The last hybrid methodology combined WSM with Pareto optimization was used to simultaneously optimize multiple objective functions. In this study, Pareto optimal solutions called noninferior solutions were generated by solving a single objective problem which was the weighted sum of the multiple objective functions (Doug & Marc, 1988); therefore, these two MCDM techniques cooperated with each other to generate the Pareto optimal set.

The conclusions can be reached based on the analysis of the three hybrid methodologies. The common characteristics among these three hybrid methodologies can be summarized into two perspectives. The first characteristic was that the Pareto optimization was just one part of the decision making process which meant it did not stay in a dominant role, but only contributed one part to the whole decision making process. And the second common characteristic was that Pareto optimization in these three hybrid methodologies were all used to generate Pareto optimal set. Consequently, the Pareto optimization was a good MCDM method to prioritize the candidates during the process of decision making.

5.3.3 Comparison between AHP and Pareto Optimization

Since data have been extracted based on the first research question, a ranking of MCDM methods has been generated based on the frequency of their utilization in different primary studies. The first two MCDM methods are AHP and Pareto optimization, the frequency of their utilization are 15 and 9, and there are 56 primary studies in all, which means almost half of the primary studies utilized them in their processes of decision making. This section is going to respectively summarize their context of utilization and discuss the differences between these two MCDM methods based on the summary.

AHP and Contexts

AHP has been discussed so much in section 5.3.1 and 5.3.2, so in this section, it will restate the characteristics of AHP and make a conclusion of its context. AHP (Saaty, 1980) is a popular MCDM method which helps decision makers to resolve the multi-criteria problems by decomposing them into a hierarchical model (Norita & Phillip, 2006; Ying-Fu & Ming-Hui, 2010). The context of AHP's utilization was concluded like that the MCDM problems have multiple criteria and multiple alternatives, there is only one specific goal which is needed to be achieved by considering all the criteria and alternatives, and all the criteria can have weights which assigned by decision makers. Most of the MCDM problems which applied AHP are about the selection of alternatives (P15, P74, P37, P48, P51, P41, P24, P70, and P71), because this kind of problem is easy to decompose into a hierarchy, even though it seems to be so complicated, but once the hierarchy is built, then the whole decision making process would be clear and simple.

Pareto Optimization and Contexts

Pareto optimization which is also called multi-objective optimization is a multiple criteria decision making method which states that a solution is optimal if it is impossible to find a solution which improves on one or more of the objectives without worsening any of them (Rose, Clarence, Marcelo, Jim, & Henk, 2005). There are 9 primary studies which have applied Pareto optimization in their studies, the context of Pareto optimization's utilization will be summarized from those studies. This MCDM method was used to explore design spaces for embedded system in primary study 14 and 2. The simultaneous consideration of several incomparable and often competing objectives were involved in the process of optimization which attempts to optimize conflicting criteria, once the set of Pareto-optimal design points was identified from a multi-objective design space exploration, the system designer had to select a design point which described a tradeoff between the various objective functions (Alessandro, Maurizio, & Davide, 2006; Colin, Cristinel, & Romeo, 2008). Primary study 19 also identified their Pareto-optimal design point by the exploration of the design space, and in the objective space, the outcomes of each alternative for each criteria formed a Pareto-optimal design point (Emmanuel, David, & Earl, 2014).

Pareto-optimal set can also be called Pareto frontier which describes the tradeoffs between objectives, so it comprises a set of candidates which are superior in all objectives, from which a final optimal design should be chosen (Aniela, Christopher, & Achille, 2003; Fatah, Kash, & Andrea, 2012). While the process of using Pareto frontier which included computing the Pareto frontier for multi-objective problems, determining the set of solutions representing the frontier, and performing a sensitivity analysis of the frontier is difficult (Fatah, Kash, & Andrea, 2012), so it needed decision makers to explore this set, the Pareto frontier, to help them to identify what can be achieved and to select one alternative which is in that set that corresponds to an appropriate tradeoff between

multiple objectives (William & Emmanuel, 2011). Since MCDM problems are always with multiple conflicting criteria or objectives, one approach to solve them is to transform them into a set of single criterion or objective problems (Colin, Cristinel, & Romeo, 2008; William & Emmanuel, 2011). One advantage of applying the Pareto optimization is that more data point for each heuristic over a data set can be used for searching the optimal solutions, rather than one data point only for each heuristic (Jingzhou & Guenther, 2008).

After summarizing all the characteristics of Pareto optimization from primary studies, the contexts which Pareto optimization would be utilized were described as that the MCDM problems should be multiple criteria or objectives. And if a large set of data were available for each alternative and a great number of alternatives were also available simultaneously, Pareto optimization can be also applied (Jingzhou & Guenther, 2008). Software engineers usually utilize Pareto optimization in the phase of analysis and design (P2, P7, P14, P18, P20, P31, and P69), because it relates to so many design objectives with multiple conflicting criteria, and Pareto optimization can resolve those tradeoffs among them with the Pareto frontier, then help decision makers make the best selection.

Comparison

Since AHP and Pareto optimization are the two most frequent used MCDM methods in our systematic literature review, the comparison between them will be conducted based on the operational usefulness which can be (1) ease of use by inexperienced decision makers, (2) the clearness of the method's logic to the decision maker, and (3) without ambiguity regarding the clarification of inputs obtained from the decision maker (Stewart, 1992). On the aspect of the first assessment criterion, AHP is much simpler to use than Pareto optimization, because its hierarchical model is clear, and the pairwise comparison of AHP can be conducted by anyone without professional relevant knowledge of decision making.

With regard to the clearness of the method's logic, AHP is also easier to understand than Pareto optimization, Pareto optimization relates to many mathematical algorithms, and the optimization process needs to be carried on by MATLAB which is a professional tool for numerical computing. For the third assessment criterion, AHP and Pareto optimization both don't have ambiguity on its inputs, because the inputs are different criteria and alternatives, they should be clear before the start of decision making. Those criteria and alternatives just need to be assigned to the predefined hierarchy model if AHP were selected, while if Pareto optimization were selected, those criteria would be transformed into Pareto optimal set. Through the comparison between AHP and Pareto optimization based on three assessment criteria, a result can be reached that AHP is more acceptable for most of the decision makers, Pareto optimization is suitable for the complex research.

6. Conclusions

The objectives of this thesis are about the investigations of the current situations of MCDM in software development process. To fulfill those objectives, three research questions were derived. In order to answer those research questions, numbers of evidences were extracted from 56 primary studies which were selected after a long time process of SLR. The first research question is about the software development problems addressed by MCDM techniques. 33 types of software development problems were identified from those primary studies. And they were classified into six groups based on the commonalities among problems. The most frequent occurred problem group is the group of software methods which involves the selection of tools, software architecture styles, configuration items, algorithms, alternative systems, and other relevant problems. The most occurred software development problems are components selection, design concepts selection and performance evaluation, and each of them got 8.9% primary studies involved.

The second research question is about the distributions of MCDM problems in software development process phases. The results show that 55.4% MCDM problems occurred in software design phase, while in software implementation and validation phase, few MCDM problems occurred. The last research question is about the utilization of MCDM techniques in software development. As a result, 33 types of MCDM techniques were identified from the primary studies, and the most frequent used MCDM techniques are AHP and Pareto optimization, which 26.8% and 17.9% primary studies are respectively involved. Besides them, TOPSIS, Weighted Sum Model, MAUT, OWA and QFD are also commonly used by decision makers in software development. In some primary studies, decision makers combined different MCDM methods into a new hybrid MCDM methodology, and most of the methods were combined with AHP or TOPSIS.

The main contributions of this thesis are that it is able to provide the references when decision makers want to select the appropriate technique to cope with the MCDM problems in software development. The decision contexts of AHP and Pareto optimization have been also discussed, which are the most frequent used MCDM methods, so it is helpful for decision makers to compare the decision situations when they intend to use those two techniques. Numbers of limitations influence the scientific values of this thesis. For example, in the stage of data extraction, some equivocal problems were decided by without consulting with the other members who also participated in this study. In the stage of data classification, many issues are ambiguity for me because of my limited knowledge of software development process phases, I had to learn them by reading some literature, so there might be some misjudgments about the classification of software development problems. The biggest limitation of our study might be the long duration of this study, it is over 10 months, so the literature was searched ten months ago might not fully represent the state-of-the-art of MCDM in software development. Besides them, numbers of small factors will also impact the results of our study.

The recommended future researches will focus on the exploration on the Pareto optimization, because it is hard to understand and most of them are mixed with other MCDM methods, i.e. they are always one part of other method, so inventing a unique set of Pareto optimization steps is crucial for decision makers to apply them. Besides that, a table of MCDM techniques and its corresponding contexts should be summarized and concluded, which like a MCDM technique dictionary or a handbook for decision makers in software development.

References

Ahmad, J. B., Mohsen, F. R., & Farshad, A. (2010). Using fuzzy multiple criteria decision making In evaluation of software quality. *Computational Intelligence and Software Engineering (CiSE), 2010 International Conference on* (pp. 1-7). Wuhan: IEEE.

Akao, Y. (1994). Development History of Quality Function Deployment. *The Customer Driven Approach to Quality Planning and Deployment*, 339.

Alessandro, G. D., Maurizio, P., & Davide, P. (2006). Fuzzy Decision Making in Embedded System Design. *Hardware/Software Codesign and System Synthesis, 2006. CODES+ISSS '06. Proceedings of the 4th International Conference* (pp. 223-228). Seoul: IEEE.

Alexander, C., & Emmanouil, S. (2013). Combining metrics for software evolution assessment by means of Data Envelopment Analysis. *Journal of Software: Evolution and Process*, 303-324.

Andrea, D., Massimo, G., & Nathan, L. (1996). Multicriteria evaluation model for flexible manufacturing system design. *Computer Integrated Manufacturing Systems*, 171-178.

Andrew, S., & Jennifer, G. (2005). *Applied Software Project Management.* O'Reilly Media, Inc.

Aniela, M., Christopher, M., & Achille, M. (2003). Multicriteria Decision Making for Production System Conceptual Design Using s-Pareto Frontiers. *Proceedings of 44th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, (pp. 1-10). Norfolk, Virginia.

Belton, V., & Stewart, T. (2002). *Multiple Criteria Decision Analysis: an integrated approach.* Springer.

Belton, V., & Stewart, T. (2010). Problem Structuring and Multiple Criteria Decision Analysis. *International Series in Operations Research & Management Science, 142*, 209-239.

Boehm, B., Egyed, A., Kwan, J., Port, D., Shah, A., & Madachy, R. (1998). Using the Win-Win spiral model: A case study. *Computer, 31*(7), 33-44.

Bourque, P., Dupuis, R., Abran, A., Moore, J. W., & Tripp, L. (1999). The Guide to the Software Engineering Body of Knowledge. *IEEE Software*, 35-44.

Brans, J. P., & Vincke, P. (1985). A Preference Ranking Organisation Method: (The PROMETHEE Method for Multiple Criteria). *Management Science, 31*(6), 647-656.

Büyüközkan, G., & Ruan, D. (2008). Evaluation of software development projects using a fuzzy multi-criteria decision approach. *Mathematics and Computers in Simulation, 77*(5-6), 464-475. doi:10.1016/j.matcom.2007.11.015

Carletta, J. (1996). Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics*, 249-254.

Chi-Yo, H., Hsiang-Chun, L., Gwo-Hshiung, T., & Hong-Yuh, L. (2010). Configuring an Embedded System by the Concepts of Emotional Design by Using a Non-additive Fuzzy Integral Based FMCDM Framework. *Technology Management for Global Economic Growth (PICMET), 2010 Proceedings of PICMET '10* (pp. 1-7). Phuket: IEEE.

Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 37-46.

Colin, A. B., Cristinel, M., & Romeo, P. G. (2008). Concept Design for Transmission Systems. *International Multi-Conference on Engineering and Technological Innovation: IMETI 2008.* Orlando, Florida, USA.

Cooper, K., Joao, W. C., & Eric, W. (2007). An Architectural Framework for the Design and Analysis of Autonomous Adaptive Systems . *Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International* (pp. 268-278). Beijing: IEEE.

Divya, K., & Misra, A. K. (2013). Software Development Cost Estimation Using Similarity Difference between Software Attributes . *Proceedings of the 2013 International Conference on Information Systems and Design of Communication* (pp. 1-6). New York: ACM.

Dodgson, J. S., Spackman, M., Pearman, A., & Phillips, L. D. (2009). *Multi-criteria analysis: a manual.* London: Department for Communities and Local Government. Retrieved from http://www.communities.gov.uk/

Doug, K., & Marc, B. (1988). A Parametric LQ Approach to Multiobjective Control System Design. *Proceedings of the 27th Conference on Decision and Control* (pp. 1278-1284). Austin, Texas: IEEE.

Eldrandaly, K., Ahmed, A. H., & AbdelAziz, N. (2009). An expert system for choosing the suitable MCDM method for solving a spatial decision problem. *Proceedings of the 9th lnternational Conference on Production Engineering, Design and Control.* Alexandria, Egypt: PEDAC.

Emmanuel, L., David, S., & Earl, T. B. (2014). Uncertainty, Risk, and Information Value in Software Requirements and Architecture. *Proceedings of the 36th International Conference on Software Engineering* (pp. 883-894). Hyderabad, India: ACM.

Falessi, D., Cantone, G., Kazman, R., & Kruchten, P. (2011). Decision-making techniques for software architecture design: A comparative survey. *ACM Computing Surveys (CSUR), 43*(4).

Fatah, C., Kash, B., & Andrea, S. V. (2012). Sensitivity analysis for simulation-based decision making: Application to a hospital emergency service design. *Simulation Modelling Practice and Theory*, 99-111.

Fishburn, P. C. (1967). Additive Utilities with Incomplete Product Sets: Application to Priorities and Assignments. *Operations Research*, 537-542.

Gaetana, S., Toberiu, S., & Ivica, C. (2013). Partitioning Decision Process for Embedded Hardware and Software Deployment. *Proceedings of 2013 IEEE 37th Annual Computer Software and Applications Conference Workshops* (pp. 674-680). Japan: IEEE.

Geldermann, J., & Rentz, O. (2001). Integrated technique assessment with imprecise information as a support for the identification of best available techniques (BAT). *OR-Spektrum, 23*(1), 137-157.

Grünbacher, P., & Seyff, N. (2005). Requirements Negotiation. *Engineering and Managing Software Requirements*, 143-162.

Gwo-Hshiung, T., & Jih-Jeng, H. (2011). *Multiple Attribute Decision Making: Methods and Applications.* CRC Press.

Gwo-Hshiung, t., Tzay-An, s., & Chien-Yuan, l. (1992). Application of multicriteria decision making to the evaluation of new energy system development in Taiwan. *Energy*, 983-992.

Gülçin, B., & Da, R. (2008). Evaluation of software development projects using a fuzzy multi-criteria decision approach. *Mathematics and Computers in Simulation*, 464-475.

Hing-Hsue, C., Jing-Rong, C., & Tien-Hwa, H. (2006). Dynamic fuzzy OWA model for evaluating the risks of software development. *Cybernetics and Systems: An International Journal*, 791-813.

Hwang, C. L., & Masud, A. S. (1979). *Multiple objective decision making, methods and applications: a state-of-the-art survey.* Berlin Heidelberg: Springer-Verlag.

Hwang, C.-L., & Yoon, K. (1981). *Multipe Attribute Decision Making: Methods and Applications.* New York: Springer-Verlag.

Jacobson, I., Booch, G., & Rumbaugh, J. (1999). *The Unified Software Development Process.* Addison-Wesley Professional.

Jett, D. B., & Midkiff, S. F. (1995). ESTEVAL: a decision support tool for hardware/software system design. *Systems Engineering of Computer Based Systems, 1995, Proceedings of the 1995 International Symposium and Workshop on* (pp. 315-322). Tucson, AZ, USA: IEEE.

Jingzhou, L., & Guenther, R. (2008). Multi-criteria decision analysis for customization of estimation by analogy method AQUA+. *Proceedings of the 4th international workshop on Predictor models in software engineering* (pp. 55-62). New York: ACM.

Jiunn-Chenn, L., Taho, Y., & Cheng-Yi, W. (2010). A lean pull system design analysed by value stream mapping and multiple criteria decision-making method under demand uncertainty. *International Journal of Computer Integrated Manufacturing*, 211-228.

José, F., Salvatore, G., & Matthias, E. (2005). *Multiple Criteria Decision Analysis: State of the Art Surveys* (Vol. 78). Springer.

Juite, W., & Yung-I, L. (2003). A fuzzy multicriteria group decision making approach to select configuration items for software development. *Fuzzy Sets and Systems*, 343-363.

Kenia, S., Hildeberto, M., & Elizabeth, F. (2006). Applying a multi-criteria approach for the selection of usability patterns in the development of DTV applications. *Proceedings of VII Brazilian symposium on Human factors in computing systems* (pp. 91-100). Natal, Brazil: ACM.

Kenneth, C., & Tom, E. (2001). *What is Operations Research*. Retrieved from HSOR.org: http://hsor.org/what_is_or.cfm?name=mutli-attribute_utility_theory

Khan, M. A., Azra, P., & Mohd, S. (2014). A Method for the Selection of Software Development Life Cycle Models using Analytic Hierarchy Process. *Proceedings of 2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)* (pp. 534-540). Ghaziabad: IEEE.

Kitchenham, B. (2004). *Procedure for performing systematic reviews.* Keele, UK: Keele University.

Künzli, S., Lothar, T., & Eckart, Z. (2005). Modular design space exploration framework for embedded systems. *IEE Proceedings-Computers and Digital Techniques*, (pp. 183-192).

Laarhoven, P. v., & Pedrycz, W. (1983). A fuzzy extension of Saaty's priority theory. *Fuzzy Sets and Systems*, 199-227.

Labaree, L. W., & Bell, J. W. (1956). *Mr. Franklin: A selection from his personal letters.* New Haven: Yale University Press.

Lamata, M. T. (2004). Ranking of alternatives with ordered weighted averaging operators. *International Journal of Intelligent Systems*, 473-482.

Li, J., & Armin, E. (2003). Decision support for requirements engineering process development . *Electrical and Computer Engineering, 2003. IEEE CCECE 2003. Canadian Conference on* (pp. 1359-1362). Montreal, Quebec: IEEE.

Macharis, C., Springael, J., De Brucker, K., & Verbeke, A. (2004). PROMETHEE and AHP: The design of operational synergies in multicriteria analysis. Strengthening PROMETHEE with ideas of AHP. *European Journal of Operational Research, 153*(2), 307−317.

Malczewski, J. (2006). GIS-based multicriteria decision analysis: a survey of the literature. *International Journal of Geographical Information Science, 20*(7), 703-726.

Marina, P., Jocelyn, S., & Hernán, A. (2014). Semi-automated Tool Recommender for Software Development Processes. *Electronic Notes in Theoretical Computer Science*, 95-109.

Mateo, J. R. (2012). *Multi Criteria Analysis in the Renewable Energy Industry.* London: Springer.

Miettinen, K. (1998). *Nonlinear Multiobjective Optimization.* Springer.

Mindy, L., Hoang, P., & Xuemei, Z. (1999). A methodology for priority setting with application to software development process. *European Jounal of Operational Research*, 375-389.

Muhamad, M. R. (1997). The deployment of strategic requirements in manufacturing systems design . *Factory 2000 - The Technology Exploitation Process, Fifth International Conference on (Conf. Publ. No. 435)* (pp. 478-485). Cambridge: IET.

Mumin, H. (2012). A Fuzzy Multi Criteria Decision Making Approach to Software Life Cycle Model Selection. *2012 38th Euromicro Conference on Software Engineering and Advanced Applications* (pp. 384-391). Cesme, Izmir, Turkey: IEEE.

Nicolás, J., & Toval, A. (2009). On the generation of requirements specifications from software engineering models: A systematic literature review. *Information and Software Technology*, 1291-1307.

Nicolò, P., Carmine, G., Michael, U., & Tony, G. (2014). Software development in startup companies: A systematic mapping. *Information and Software Technology*, 1200-1218.

Niekamp, R. (2005). Software Component Architecture. *Gestión de Congresos - CIMNE/Institute for Scientific Computing*, 4.

Norita, A., & Phillip, A. L. (2006). Software Project Management Tools: Making a Practical Decision Using AHP . *Software Engineering Workshop, 2006. SEW '06. 30th Annual IEEE/NASA* (pp. 76-84). Columbia, MD: IEEE.

Omolade, S., & Guenther, R. (2005). Software release planning for evolving systems. *Innovations in System and Software Engineering*, 189-204.

Paul, R., & Yair, W. (2009). A Proposal for a Formal Definition of the Design Concept. *Design Requirements Engineering: A Ten-Year Perspective*, 103-136.

Pierre, C., Mambaye, L., Abdelhak, I., Vincent, C., & Jacky, M. (2014). Tracking the consequences of design decisions in mechatronic Systems Engineering. *Mechatronics*, 763-774.

Pressman, R. S. (2005). *Software Engineering: A Practitioner's Approach.* Palgrave Macmillan.

Radatz, J., Geraci, A., & Katki, F. (1990). *IEEE standard glossary of software engineering terminology.* IEEE Std.

Raffo, D. M., Harrison, W., & Joseph, V. (2002). Software Process Decision Support: Making Process Tradeoffs Using a Hybrid Metrics, Modeling and Utility Framework. *Proceedings of the 14th international conference on Software engineering and knowledge engineering* (pp. 803-809). Ischia Island, Italy: ACM.

Rajabally, E. S., & Holywell, P. (2006). Using Fuzzy Decision Support to Compare Systems Modelling Tools. *INCOSE International Symposium*, 1557–1569.

Rose, X. L., Clarence, W. d., Marcelo, H. A., Jim, A. N., & Henk, C. (2005). A New Approach for Mechatronic System Design: Mechatronic Design Quotient (MDQ). . *Proceedings of the 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics* (pp. 911-915). Monterey, California: IEEE.

Roubens, M. (1982). Preference Relations an Actions and Criteria in Multicriteria Decision. *European Journal of Operational Research, 10*(1), 51-55.

Ruhe, G., & Saliu, M. O. (2005). The Art and Science of Software Release Planning. *Software, IEEE, 22*(6), 47-53. doi:10.1109/MS.2005.164

Saaty, T. L. (1980). *The Analytic Hierarchy Process: Planning, Priority Setting, Resource Allocation.* New York: McGraw-Hill.

Sommerville, I. (2004). *Software Engineering* (7th ed.). Pearson addition Wesley.

Stewart, T. J. (1992). A critical survey on the status of multiple criteria decision making theory and practice. *Omega, 20*(5-6), 569-586.

Suryanarayana, G., Samarthyam, G., & Sharma, T. (2014). *Refactoring for Software Design Smells: Managing Technical Debt.* Morgan Kaufmann.

Taguchi, G. (1993). *Taguchi on Robust Technology Development: Bringing Quality Engineering Upstream.* The American Society of Mechanical Engineers. doi:10.1115/1.800288

Thomas, N., & Christian, S. (2007). Interactive Decision Support for Multiobjective COTS Selection. *Proceedings of the 40th Hawaii International Conference on System Sciences* (pp. 283b-283b). Hawaii: IEEE.

Ullah, R., De-Qun, Z., Peng, Z., Mukarrum, H., & Amjad, S. M. (2013). An approach for space launch vehicle conceptual design and multi-attribute evaluation. *Aerospace Science and Technology*, 65-74.

Wallenius, J., & Zionts, S. (2011). *Multiple Criteria Decision Making: From Early History to the 21st Century.* World Scientific.

Vassilis, C. G., & Pandelis, G. I. (2007). Multi Objective Analysis for Timeboxing Models of Software Development. *Second International Conference on Software and Data Technologies* (pp. 145-153). Barcelona: IEEE.

William, H., & Emmanuel, L. (2011). Simulating and Optimising Design Decisions in Quantitative Goal Models. *Proceedings of 2011 IEEE 19th International Requirements Engineering Conference* (pp. 79-88). Trento: IEEE.

Vinzent, R., Jürgen, G., & Gerald, R. (2014). Assessment of production system alternatives during early development phase. *2014 Conference on Systems Engineering Research* (pp. 34-43). Redondo Beach, California: ELSEVIER.

Woodside, M., Carleton, U. O., Franks, G., & Petriu, D. (2007). The future of software performance engineering. *Future of Software Engineering, 2007. FOSE '07* (pp. 171-187). Minneapolis, MN: IEEE. doi:10.1109/FOSE.2007.32

Xiaoqian, S., & Yongchang, L. (2010). Evaluation of control system performance using Multiple Criteria Decision Making techniques. *Decision and Control (CDC), 2010 49th IEEE Conference* (pp. 3718-3723). Atlanta: IEEE.

Yager, R. R. (1993). Families of OWA operators. *Fuzzy Sets and Systems*, 125-148.

Yager, R. R. (1996). Constrained OWA aggregation. *Fuzzy Sets and Systems*, 89-101.

Yi, P., Gang, K., Guoxun, W., Wenshuai, W., & Honggang, W. (2010). Evaluating software reliability: Integration of MCDM and data mining. *Software Engineering and Data Mining (SEDM), 2010 2nd International Conference on* (pp. 613-617). Chengdu: IEEE.

Ying-Fu, L., & Ming-Hui, W. (2010). A fuzzy-AHP-based technique for the decision of design feature selection in Massively Multiplayer Online Role-Playing Game development. *Expert Systems with Applications*, 8685-8693.

Yoon, K. (1987). A Reconciliation Among Discrete Compromise Solutions. *Journal of the Operational Research Society*, 277-286.

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, 338-353.

Appendix A. Search strings in different databases

IEEE Xplore:

("Document Title": "decision-making" OR "Document Title": "making decisions" OR "Document Title": “decision analysis” OR "Document Title": "decision support" OR "Abstract": "decision-making" OR "Abstract": "making decisions" OR "Abstract": "decision support" OR "Abstract": “decision analysis” OR "Author Keywords": "decision-making" OR "Author Keywords": "making decisions" OR "Author Keywords": "decision support" OR "Author Keywords": “decision analysis”) AND ("Document Title": "software development" OR "Document Title": "software design" OR "Document Title": "software engineering" OR "Document Title": "system development" OR "Document Title": "system design" OR "Document Title": "system engineering" OR "Abstract": "software development" OR "Abstract": "software design" OR "Abstract": "software engineering" OR "Abstract": "system development" OR "Abstract": "system design" OR "Abstract": "system engineering" OR "Author Keywords": "software development" OR "Author Keywords": "software design" OR "Author Keywords": "software engineering" OR "Author Keywords": "system development" OR "Author Keywords": "system design" OR "Author Keywords": "system engineering")

ACM:

(Title: "decision-making" OR Title: "making decisions" OR Title: "decision support" OR Title: "decision analysis" OR Abstract: "decision-making" OR Abstract: "making decisions" OR Abstract: "decision support" OR Abstract: "decision analysis" OR Keywords: "decision-making" OR Keywords: "making decisions" OR Keywords: "decision support" OR Keywords: "decision analysis") AND (Title: "software development" OR Title: "software design" OR Title: "software engineering" OR Title: "system development" OR Title: "system design" OR Title: "system engineering" OR Abstract: "software development" OR Abstract: "software design" OR Abstract: "software engineering" OR Abstract: "system development" OR Abstract: "system design" OR Abstract: "system engineering" OR Keywords: "software development" OR Keywords: "software design" OR Keywords: "software engineering" OR Keywords: "system development" OR Keywords: "system design" OR Keywords: "system engineering")

Science Direct:

TITLE-ABSTR-KEY("decision-making" OR "making decisions" OR "decision support" OR “decision analysis”) and TITLE-ABSTR-KEY("software development" OR "software design" OR "software engineering" OR "system development" OR "system design" OR "system engineering")

Scopus:

(TITLE-ABS-KEY("decision-making" OR "making decisions" OR "decision support" OR "decision analysis") AND TITLE-ABS-KEY("software development" OR "software design" OR "software engineering" OR "system development" OR "system design" OR "system engineering"))

Web of Science:

TOPIC: (("decision-making" OR "making decisions" OR "decision support" OR "decision analysis") AND ("software development" OR "software design" OR "software engineering" OR "system development" OR "system design" OR "system engineering"))

ProQuest:

(ab("decision-making" OR "making decision" OR "decision support" OR "decision analysis") AND ab("software development" OR "software design" OR "software engineering" OR "system development" OR "system design" OR "system engineering")) OR(ti("decision-making" OR "making decision" OR "decision support" OR "decision analysis") AND ti("software development" OR "software design" OR "software engineering" OR "system development" OR "system design" OR "system engineering")) OR (if("decision-making" OR "making decision" OR "decision support" OR "decision analysis") AND if("software development" OR "software design" OR "software engineering" OR "system development" OR "system design" OR "system engineering"))

Appendix B. The references of Primary Studies

No.	Reference
P1	Ahmad, J. B., Mohsen, F. R., and Farshad, A. (2010). Using fuzzy multiple criteria decision making In evaluation of software quality. *Computational Intelligence and Software Engineering (CiSE), 2010 International Conference on* (pp. 1-7). Wuhan: IEEE.
P2	Alessandro, G. D., Maurizio, P., and Davide, P. (2006). Fuzzy Decision Making in Embedded System Design. *Hardware/Software Codesign and System Synthesis, 2006. CODES+ISSS '06. Proceedings of the 4th International Conference* (pp. 223-228). Seoul: IEEE.
P3	Alexander, C., and Emmanouil, S. (2013). Combining metrics for software evolution assessment by means of Data Envelopment Analysis. *Journal of Software: Evolution and Process*, 303-324.
P4	Ana, B., and Gilberto, M. (2006). Supporting the allocation of software developmentwork in distributed teams with multi-criteria decision analysis. *The International Jounal of management science*, 464-475.
P5	Andrea, D., Massimo, G., and Nathan, L. (1996). Multicriteria evaluation model for flexible manufacturing system design. *Computer Integrated Manufacturing Systems*, 171-178.
P6	Andrew, O., and Kemper, L. (2006). A decision support framework for flexible system design. *Journal of Engineering Design*, 75-97.
P7	Aniela, M., Christopher, M., and Achille, M. (2003). Multicriteria Decision Making for Production System Conceptual Design Using s-Pareto Frontiers. *Proceedings of 44th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, (pp. 1-10). Norfolk, Virginia.
P8	Ansgar, L., Jürgen, M., and Dieter, R. (2009). A Decision Model for Supporting Task Allocation Processes in Global Software Development. *Proceedings of the 10th International Conference on Product- Focused Software Process Improvement (PROFES 2009)* (pp. 332-346). Oulu, Finland: Springer.
P12	Chi-Yo, H., Hsiang-Chun, L., Gwo-Hshiung, T., and Hong-Yuh, L. (2010). Configuring an Embedded System by the Concepts of Emotional Design by Using a Non-additive Fuzzy Integral Based FMCDM Framework. *Technology Management for Global Economic Growth (PICMET), 2010 Proceedings of PICMET '10:* (pp. 1-7). Phuket: IEEE.
P14	Colin, A. B., Cristinel, M., and Romeo, P. G. (2008). Concept Design for Transmission Systems. *International Multi-Conference on Engineering and Technological Innovation: IMETI 2008.* Orlando, Florida, USA.
P15	Cooper, K., Joao, W. C., and Eric, W. (2007). An Architectural Framework for the Design and Analysis of Autonomous Adaptive Systems . *Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International* (pp. 268-278). Beijing: IEEE.
P16	Divya, K., and Misra, A. K. (2013). Software Development Cost Estimation Using Similarity Difference between Software Attributes. *Proceedings of the 2013 International Conference on Information Systems and Design of Communication* (pp. 1-6). New York: ACM.
P18	Doug, K., and Marc, B. (1988). A Parametric LQ Approach to Multiobjective Control System Design. *Proceedings of the 27th Conference on Decision and Control* (pp. 1278-1284). Austin, Texas: IEEE.

P19	Emmanuel, L., David, S., and Earl, T. B. (2014). Uncertainty, Risk, and Information Value in Software Requirements and Architecture. *Proceedings of the 36th International Conference on Software Engineering* (pp. 883-894). Hyderabad, India: ACM.
P20	Fatah, C., Kash, B., and Andrea, S. V. (2012). Sensitivity analysis for simulation-based decision making: Application to a hospital emergency service design. *Simulation Modelling Practice and Theory*, 99-111.
P21	Gaetana, S., Tiberiu, S., and Ivica, C. (2013). Partitioning Decision Process for Embedded Hardware and Software Deployment. *Proceedings of 2013 IEEE 37th Annual Computer Software and Applications Conference Workshops* (pp. 674-680). Japan: IEEE.
P22	Golnaz, E., and Eric, Y. (2011). A Semi-Automated Decision Support Tool for Requirements Trade-off Analysis. *2011 35th IEEE Annual Computer Software and Applications Conference* (pp. 466-475). Munich: IEEE.
P24	GWO-HSHIUNG, T., TZAY-AN, S., and CHIEN-YUAN, L. (1992). Application of multicriteria decision making to the evaluation of new energy system development in Taiwan. *Energy*, 983-992.
P25	Gülçin, B., and Da, R. (2008). Evaluation of software development projects using a fuzzy multi-criteria decision approach. *Mathematics and Computers in Simulation*, 464-475.
P26	Hing-Hsue, C., Jing-Rong, C., and Tien-Hwa, H. (2006). DYNAMIC FUZZY OWA MODEL FOR EVALUATING THE RISKS OF SOFTWARE DEVELOPMENT. *Cybernetics and Systems: An International Journal*, 791-813.
P28	In, H. P., David, O., and Tom, R. (2002). Multi-Criteria Preference Analysis for Systematic Requirements Negotiation. *Proceedings of Computer Software and Applications Conference* (pp. 887-892). Oxford, England: IEEE.
P29	Jett, D. B., and Midkiff, S. F. (1995). ESTEVAL: a decision support tool for hardware/software system design. *Systems Engineering of Computer Based Systems, 1995., Proceedings of the 1995 International Symposium and Workshop on* (pp. 315-322). Tucson, AZ, USA: IEEE.
P31	Jingzhou, L., and Guenther, R. (2008). Multi-criteria decision analysis for customization of estimation by analogy method AQUA+. *Proceedings of the 4th international workshop on Predictor models in software engineering* (pp. 55-62). New York: ACM.
P32	Jiunn-Chenn, L., Taho, Y., and Cheng-Yi, W. (2010). A lean pull system design analysed by value stream mapping and multiple criteria decision-making method under demand uncertainty. *International Journal of Computer Integrated Manufacturing*, 211-228.
P35	Juite, W., and Yung-I, L. (2003). Afuzzy multicriteria group decision making approach to select configuration items for software development. *Fuzzy Sets and Systems*, 343-363.
P36	Kenia, S., Hildeberto, M., and Elizabeth, F. (2006). Applying a multi-criteria approach for the selection of usability patterns in the development of DTV applications. *Proceedings of VII Brazilian symposium on Human factors in computing systems* (pp. 91-100). Natal, Brazil: ACM.
P37	Khan, M. A., Azra, P., and Mohd, S. (2014). A Method for the Selection of Software Development Life Cycle Models using Analytic Hierarchy Process. *Proceedings of 2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)* (pp. 534-540). Ghaziabad: IEEE.
P39	Künzli, S., Lothar, T., and Eckart, Z. (2005). Modular design space exploration framework for embedded systems. *IEE Proceedings-Computers and Digital Techniques*, 183-192.

P40	Li, J., and Armin, E. (2003). Decision support for requirements engineering process development . *Electrical and Computer Engineering, 2003. IEEE CCECE 2003. Canadian Conference on* (pp. 1359-1362). Montreal, Quebec: IEEE.
P41	Marina, P., Jocelyn, S., and Hernán, A. (2014). Semi-automated Tool Recommender for es. *Electronic Notes in Theoretical Computer Science*, 95-109.
P43	Mindy, L., Hoang, P., and Xuemei, Z. (1999). A methodology for priority setting with application to . *European Jounal of Operational Research*, 375-389.
P44	Mirosław, D., and Grzegorz, G. (2008). Multi-CriterionEvaluation of Development Strategy Components in the Presence of Intangibles and Uncertainty. *19th International Conference on Systems Engineering* (pp. 464-467). IEEE.
P45	Moaven, S., Habibi, J., Ahmadi, H., and Kamandi, A. (2008). A Decision Support System for Software Architecture-Style Selection. *Software Engineering Research, Management and Applications, 2008. SERA '08. Sixth International Conference on* (pp. 213-220). Prague: IEEE.
P46	Mohammad, R. H., Nasser, H., and Peter, J. W. (2008). A Multi-Criteria Decision Framework for Optimal Augmentation of Transmission Grid-Addressing a Tool for Sensitive Zone Detection in Electricity Market. *International Journal of Emerging Electric Power Systems*, 1-16.
P47	Muhamad, M. R. (1997). The deployment of strategic requirements in manufacturing systems design . *Factory 2000 - The Technology Exploitation Process, Fifth International Conference on (Conf. Publ. No. 435)* (pp. 478-485). Cambridge: IET.
P48	Mumin, H. (2012). A Fuzzy Multi Criteria Decision Making Approach to Software Life Cycle Model Selection. *2012 38th Euromicro Conference on Software Engineering and Advanced Applications* (pp. 384-391). Cesme, Izmir, Turkey: IEEE.
P51	Norita, A., and Phillip, A. L. (2006). Software Project Management Tools: Making a Practical Decision Using AHP. *Software Engineering Workshop, 2006. SEW '06. 30th Annual IEEE/NASA* (pp. 76-84). Columbia, MD : IEEE.
P53	Omolade, S., and Guenther, R. (2005). Software release planning for evolving systems. *Innovations in Systems and Software Engineering*, 189-204.
P54	Pierre, C., Mambaye, L., Abdelhak, I., Vincent, C., and Jacky, M. (2014). Tracking the consequences of design decisions in mechatronic Systems Engineering. *Mechatronics*, 763-774.
P55	Raffo, D. M., Harrison, W., and Joseph, V. (2002). Software Process Decision Support: Making Process Tradeoffs Using a Hybrid Metrics, Modeling and Utility Framework. *Proceedings of the 14th international conference on Software engineering and knowledge engineering* (pp. 803-809). Ischia Island, Italy: ACM.
P56	Rajabally, E., Steve, W., and Holywell, P. (2006). Using Fuzzy Decision Support to Compare Systems Modelling Tools. *INCOSE International Symposium*, 1557–1569.
P58	Rose, X. L., Clarence, W. d., Marcelo, H. A., Jim, A. N., and Henk, C. (2005). A New Approach for Mechatronic System Design: Mechatronic Design Quotient (MDQ). *Proceedings of the 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics* (pp. 911-915). Monterey, California: IEEE.
P60	Ruhe, G., EBERLEIN, A., and PFAHL, D. (2003). Trade-off Analysis for Requirements Selection. *International Journal of Software Engineering and Knowledge Engineering*, 345-366.

P61	Shady, G. A., and Zarinah, M. K. (2013). Applying Software Architecture Comparison Analysis Method for a Critical System (Sacamcs) Based on Sacam (Concept and Case Study). *Middle-East Journal of Scientific Research*, 1471-1482.
P62	Shantesh, H., Manuel, J. L., Paula, F. V., and Luis, A. R. (2013). Incorporating sustainability in decision-making for medical device development. *Technology in Society*, 276-293.
P63	Shih-Tong, L., and Shih-Heng, Y. (2012). Risk Factors Assessment for Software Development Project Based on Fuzzy Decision Making. *International Journal of Information and Electronics Engineering*, 596-600.
P65	Thomas, N., and Christian, S. (2007). Interactive Decision Support for Multiobjective COTS Selection. *Proceedings of the 40th Hawaii International Conference on System Sciences* (pp. 283b-283b). Hawaii: IEEE.
P66	Thurimella, A. K., and Srini, R. (2012). On adopting multi-criteria decision-making approaches for variability management in software product lines. *Proceedings of the 16th International Software Product Line Conference* (pp. 32-35). New York: ACM.
P67	Ullah, R., De-Qun, Z., Peng, Z., Mukarrum, H., and M. Amjad, S. (2013). An approach for space launch vehicle conceptual design and multi-attribute evaluation. *Aerospace Science and Technology*, 65-74.
P68	Vassilis, C. G., and Pandelis, G. I. (2007). Multi Objective Analysis for Timeboxing Models of Software Development. *Second International Conference on Software and Data Technologies* (pp. 145-153). Barcelona: IEEE.
P69	William, H., and Emmanuel, L. (2011). Simulating and Optimising Design Decisions in Quantitative Goal Models. *Proceedings of 2011 IEEE 19th International Requirements Engineering Conference* (pp. 79-88). Trento : IEEE.
P70	Vincent, S. L., Bo, K. W., and Waiman, C. (2002). Group decision making in a multiple criteria environment: A case using the AHP in software selection. *European Jounal of Operational Research*, 134-144.
P71	Vinzent, R., Jürgen, G., and Gerald, R. (2014). Assessment of production system alternatives during early development phase. *2014 Conference on Systems Engineering Research* (pp. 34-43). Redondo Beach, California: ELSEVIER.
P72	Xiaoqian, S., and Yongchang, L. (2010). Evaluation of control system performance using Multiple Criteria Decision Making techniques . *Decision and Control (CDC), 2010 49th IEEE Conference* (pp. 3718-3723). Atlanta: IEEE.
P73	Yi, P., and et al. (2010). Evaluating software reliability: Integration of MCDM and data mining. *Software Engineering and Data Mining (SEDM), 2010 2nd International Conference on* (pp. 613-617). Chengdu: IEEE.
P74	Ying-Fu, L., and Ming-Hui, W. (2010). A fuzzy-AHP-based technique for the decision of design feature selection in Massively Multiplayer Online Role-Playing Game development. *Expert Systems with Applications*, 8685-8693.